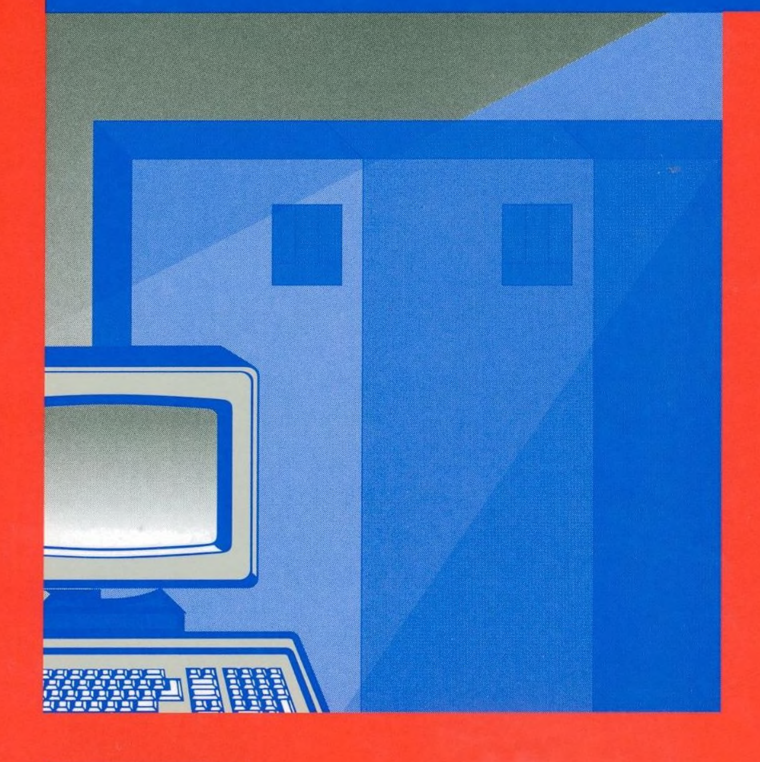
John J. Abbott

UNIX für den kommerziellen Einsatz

Die SINIX-Anwendungsumgebung





John J. Abbott

UNIX für den kommerziellen Einsatz

UNIX für den kommerziellen Einsatz

Die SINIX-Anwendungsumgebung

Herausgegeben von John J. Abbott

Mit 42 Abbildungen



Carl Hanser Verlag München Wien

Die Deutsche Bibliothek – CIP-Einheitsaufnahme

UNIX für den kommerziellen Einsatz: die SINIX-Anwendungsumgebung / hrsg. von John J. Abbott. [Autoren: Bernd Ahn . . . Überarb. und Übers. aus dem Engl.: Werner Leist-Gutmann . . .]. – München ; Wien : Hanser, 1993
ISBN 3-446-17647-0
NE: Abbott, John J. [Hrsg.]; Ahn, Bernd; Leist-Gutmann, Werner [Bearb.]

Umschlagkonzeption: Hans Peter Willberg
Druck und Bindung: Sellier Druck GmbH, Freising
Printed in Germany

Vorwort

Das vorliegende Buch erscheint aus Anlaß des zehnjährigen Bestehens des Betriebssystems SINIX — des UNIX der Siemens Nixdorf Informationssysteme AG. Es ist Ziel des Buchs, einen umfassenden Überblick über SINIX zu geben und zu zeigen, warum gerade SINIX und seine Komponenten in besonderem Maß für den kommerziellen Einsatz geeignet sind.

Meinen Dank möchte ich an dieser Stelle Klaus Gewald aussprechen, dem Leiter des Bereichs Systemtechnische Entwicklung Open Systems bei der Siemens Nixdorf Informationssysteme AG. Er war es, der die Idee zu diesem Buch hatte und seiner Förderung ist auch die erfolgreiche Verwirklichung zu verdanken. Die Arbeiten am Buch wurden koordiniert von Volker Dulich, dem Leiter der SINIX-Entwicklung und von Rainer Zimmer, unserem Vertreter bei X/Open, dem Open Systems Konsortium. Besonders verpflichtet bin ich David Ortmeyer, Valdur Koha und Eberhard Petri von der Research and Development Division, Burlington (USA), durch deren tatkräftige Unterstützung meine Arbeit erst ermöglicht wurde.

Mehr als vierzig Mitarbeiter von Siemens Nixdorf aus den Bereichen Technik, Entwicklung und Dokumentation haben Beiträge für dieses Buch geliefert. Hinter ihnen stehen die Abteilungen, welche die ganze breite Palette der SINIX-Produkte entwickeln: das Betriebssystem SINIX selbst, Werkzeuge und Arbeitsumgebungen, aber auch Anwendungsprogramme, die von SINIX unterstützt werden. Ihre Namen erscheinen in alphabetischer Reihenfolge am Ende des Vorworts.

Ich hoffe, daß es mir gelungen ist, all diese Beiträge zu einem Buch zusammenzufügen, das für Sie als Leser nicht nur interessant, sondern auch nützlich ist.

John J. Abbott

Herausgeber

Verzeichnis der Autoren: Bernd Ahn, Dieter Biehn, Holger Conradi, Hardy Dölfel, Günther Eberth, Klaus-Jürgen Eckardt, Peter Eitel, Joachim Etzrodt, Wolfgang Frielinghaus, Ruth Gansser, Lothar Gläßer, Bertram Halt, Thomas Haensse, Dietmar Hennig, Ferdinand Herrmann, Ulrich Heubach, Ingo Hoffmann, Wolfgang Horak, Bartholomaeus Kellerer, Rudolf Koch, Valdur Koha, Harald König, Burkhard Konrad, Günther Krönert, Jens Kruse, Françoise Miane, Peter Mittermaier, Jürgen Nolte, Ralf Nolting, David Ortmeyer, Jürgen Peters, Eberhard Petri, Manfred Reitenspieß, John Richardson, Norbert Richter, Michael Rust, Konrad Sassen, Walter Sattler, Dieter Schlereth, Rüdiger Schuster, Pierre Seghin, Karl Siegl, Petra Sommer, Friedrich Spörel, Manfred Unger, Jamie Wilkie, Bernd Winter, Rainer Zimmer, Ludwig Zink.

Überarbeitung und Übersetzung aus dem Englischen: Werner Leist-Gutmann, Nicole Neumayer, Martin Strübe.

Inhalt

1 Einleitung

Überblick

Das Erscheinen dieses Buches fällt in eine Zeit, in der kommerzielle Anwender versuchen, den Einsatz ihrer Computer so zu planen, daß der einmal festgelegte Weg auch in der Zukunft erfolgreich beschritten werden kann. Die Fortentwicklung der Computer-Hardware und der Netzwerke bietet die Möglichkeit, die zu erledigenden Aufgaben mit immer kleineren Computern und im Netzverbund zu lösen. Damit kommerzielle Anwender aus dieser Fortentwicklung Nutzen ziehen können, benötigen sie offene Computersysteme; denn offene Computersysteme unterstützen standardisierte Betriebssystemschnittstellen und Netzprotokolle. Damit ist sichergestellt, daß die Anwendungsprogramme portierbar sind, und die Computer im Netzverbund miteinander kommunizieren können.

Welche Bedürfnisse muß eine Firma heute bei der Anschaffung von Computersystemen berücksichtigen?

- Midrange Systeme sind heute so leistungsfähig, wie es vor kurzem noch Großrechner waren. Hochleistungs-PCs stoßen in die Leistungsbereiche der Midrange-Systeme vor.
- Unterschiedliche Computerarchitekturen, wie Mainframes (Großrechner), Server (z.B. Minicomputer), Workstations und PCs sollen zu einem Ganzen integriert werden.
- Verteilte Netzwerkarchitekturen ermöglichen es, unterschiedliche Computer in Netzwerken als dezentrale Ressourcen einzusetzen.
- Große Datenbanken und Online-Transaktionssysteme, traditionell eine Domäne der Großrechenanlagen, werden nun auf Midrange-Servern und auf Client-Server-Architekturen im Netzverbund verfügbar.

- Entwickler können leistungsfähigere Benutzerschnittstellen entwerfen, seit es Midrange-Systeme mit komfortablen Bedienoberflächen, leistungsfähigen Bildschirmen und Programmen mit Fenstertechnik gibt.
- Es soll eine einheitliche Benutzerschnittstelle vorhanden sein, mit der alle Anwendungsprogramme auf allen Plattformen bedient werden können.
- Die Möglichkeiten von modernen Benutzerschnittstellen werden gerne voll ausgenutzt. Dabei soll jedoch zusätzlich die Möglichkeit bestehen, die gleiche Anwendung auch auf weniger teuren Terminals zu betreiben, indem man lediglich eine weniger aufwendige Benutzerschnittstelle einsetzt.
- Kunden möchten die Möglichkeit haben, bei den unterschiedlichsten Software-Häusern kaufen zu können. Die heterogene Hard- und Software soll dabei leicht zu integrieren sein, denn kein Kunde will sich auf Gedeih und Verderb an einen Anbieter oder ein System binden.
- Anwender wünschen sich eine Bedienoberfläche, mit der sie Zugang zu einer breiten Palette von Ressourcen haben, seien es nun Werkzeuge zur Steigerung der individuellen Leistung, wie ein Tabellenkalkulationsprogramm, ein Textverarbeitungssystem, ein Datenbankverbund oder aber so sensible Anwendungen wie Online-Transaktionssysteme.

Dieses Buch soll Ihnen zeigen, wie Computersysteme, die auf SINIX, dem UNIX der Siemens Nixdorf Informationssysteme AG, basieren, einer Firma oder Behörde helfen können, optimal auf diese Situation zu reagieren. Dieses Einleitungskapitel soll drei einfache, aber wesentliche Fakten herausstellen:

- SINIX ist UNIX
- SINIX ist ein offenes System
- SINIX ist ideal für den kommerziellen Computereinsatz geeignet

Das Buch stellt schwerpunktmäßig spezifische Einsatzgebiete für SINIX vor. Es soll Geschäftsführern, Managern, Systemverwaltern oder Consultants Rat und Hilfe geben, denn Sie müssen entscheiden, welche Computer gekauft werden, Sie müssen Wissen vermitteln und beraten, Sie müssen Computersysteme und Netzwerke betreiben. Neben den ausführlichen Informationen über das Betriebssystem SINIX selbst gibt Ihnen das Buch einen ausgezeichneten Überblick über die aktuelle Computertechnologie, deren Standards und gängige Produkte.

SINIX ist UNIX

Als der Name SINIX geprägt wurde, stand er für das „UNIX von Siemens". Seit Zusammenschluß des Bereichs Daten- und Informationstechnik von Siemens einerseits und Nixdorf andererseits, steht SINIX für das „UNIX von Siemens Nixdorf". SINIX ist UNIX, denn es basiert auf dem weltweit gültigen Industriestandard für UNIX, dem System V Release 4 (SVR4), das der Siemens Nixdorf Informationssysteme AG von den **U**NIX **S**ystem **L**aboratories (USL) geliefert wird.

SINIX ist ein offenes System

SINIX war und ist ein offenes System, es ist eine Weiterentwicklung des klassischen offenen Systems UNIX. Die Computerabteilungen von Siemens und Nixdorf (1990 zur Siemens Nixdorf Informationssysteme AG vereinigt) unterstützten schon früh die Idee der offenen Standards: 1984 gehörten sie zu den Gründungsmitgliedern von BISON (Bull, ICL, Siemens, Olivetti und Nixdorf), einem Konsortium, aus dem nach dem Beitritt weiterer Mitglieder schließlich X/Open hervorging. X/Open, obwohl ein privates Konsortium und kein Standardisierungsgremium, ist heute die führende Organisation im Bereich der offenen Systeme. Hier wurden und werden standardisierte Schnittstellen festgelegt und veröffentlicht, die allgemein als verbindlich und von umfassender Bedeutung für offene Computersysteme anerkannt werden. Heute führt jede SINIX-Version das X/Open Warenzeichen. Es bescheinigt, daß SINIX alle Standards, die von X/Open festgelegt wurden, nachprüfbar erfüllt.

Was ist unter einem offenen System zu verstehen? Das **I**nstitute for **E**lectrical and **E**lectronics **E**ngineers (IEEE), Urheber der Standards für Portable Operating Systems Interfaces (POSIX, nunmehr in den X/Open Standard integriert), definiert ein offenes System wie folgt:

Ein offenes System ist ein System, das standardisierte Schnittstellen, Dienste und unterstützende Formate zur Verfügung stellt. Damit wird erreicht, daß konventionsgemäß entwickelte Software folgenden Leistungsmerkmalen genügt:

- Portierbarkeit über Systemgrenzen hinweg
- Kommunikationsfähigkeit mit anderen Anwendungen
- Konsistent gestaltete Bedienoberfläche für einheitliche Bedienbarkeit

Die Schnittstellen müssen dabei folgenden Anforderungen erfüllen:

- sie werden veröffentlicht
- sie werden in einem offenen Meinungsbildungsprozeß auf dem neusten Stand gehalten
- sie stehen in Übereinstimmung mit internationalen Standards

Der Verbreitung von offenen Systemen sind Anwenderorganisationen, Hersteller und Standardisierungsgremien gleichermaßen verpflichtet. Durch sie werden Standards und Systeme entwickelt, um die Abhängigkeit von einem einzelnen Anbieter zu verringern und die Flexibilität zu erhöhen.

Die Zahl der Standards, die auf Computersysteme Einfluß nehmen, ebenso wie die Liste der Organisationen, die sich in der Entwicklung und Förderung solcher Standards engagieren, ist lang. Im Anhang finden Sie eine entsprechende Übersicht.

Es sind zwei Besonderheiten, die ein offenes System auszeichnen:

- die Fähigkeit, im Netzverbund mit anderen Systemen zu kommunizieren und zusammenzuarbeiten
- die Unterstützung der unterschiedlichsten Standardschnittstellen, seien es Hardwareschnittstellen, Schnittstellen zwischen den verschiedenen Betriebssystemen oder grafische Bedienoberflächen

Sind diese Gegebenheiten erfüllt, so ist ein System als offen zu bezeichnen. Denn damit ist auch das Ziel erreicht, das mit offenen Systemen verbunden ist: Unabhängigkeit von Anbietern und den Einschränkungen proprietärer Systeme.

Wenn man von der Fähigkeit von Computern spricht, mit anderen Systemen im Netzverbund zu kommunizieren und zusammenzuarbeiten, so meint man damit, daß ein Kunde sich die unterschiedlichsten Computer von den verschiedensten Anbietern kaufen kann, die Systeme aber trotzdem miteinander sozusagen sprechen können. Dies bedeutet natürlich auch, daß Computer der unterschiedlichsten Größe und Komplexität, seien es nun Großrechner, Mehrbenutzer-Systeme, Workstations oder PCs, immer die Fähigkeit haben, untereinander Daten auszutauschen. Die Anwender dieser Computersysteme können mit E-Mail Nachrichten verschicken und Dateien austauschen.

Ein offenes Betriebssystem erfüllt idealerweise folgende Anforderungen:

- es verfügt über standardisierte und sorgfältig dokumentierte Betriebssystem-Schnittstellen für Anwendungen
- es ist nicht Eigentum einer einzigen Firma
- es ist binär und als Source verfügbar, und die Verfügbarkeit wird nicht durch Lizenzbedingungen oder überhöhte Lizenzgebühren eingeschränkt
- es ist auf einer breiten Hardware-Palette verfügbar
- es kann vom PC bis zur Großrechenanlage an die unterschiedlichsten Rechnertypen angepaßt werden
- es ist leicht auf neue Hardware-Plattformen mit den unterschiedlichsten Architekturen portierbar
- es unterstützt eine breite Palette von Peripheriegeräten
- neu entwickelte Peripheriegeräte können leicht eingebunden werden

Kein Betriebssystem kann natürlich diesen Anforderungen in allen Punkten gerecht werden. Aber UNIX ist ein Betriebssystem, das diesem Ideal schon sehr nahe kommt. SINIX, die Weiterentwicklung des Standard-UNIX von Siemens Nixdorf, ist noch einen Schritt weiter. Es hat natürlich alle Eigenschaften des offenen Systems UNIX, aber darüber hinaus wurde viel daran gesetzt, es offener als andere UNIX-Derivate zu machen.

SINIX - ideal für den kommerziellen Einsatz

Siemens Nixdorf sieht seine Marktanteile vor allem im kommerziellen Sektor. Aus diesem Grund verkaufen wir auch seit Jahren SINIX-Systeme in diesem Marktsegment. Deshalb wurde SINIX früher als andere UNIX-Derivate für den Einsatz im kommerziellen Bereich ausgebaut, um dadurch die Bedürfnisse des Marktes besser zu befriedigen. Der Markt wünscht:

- erstklassige Anwenderunterstützung mit menügesteuerten Bedienoberflächen und umfassender Online-Hilfe

- unverminderte Pflege und Weiterentwicklung der preisgünstigen alphanumerischen Terminals, da diese bei vielen Anwendungen im kommerziellen Bereich eingesetzt werden
- leistungsfähige Entwicklungsumgebungen zum Entwickeln und Testen von Anwendungsprogrammen
- hervorragende Werkzeuge zur System- und Netzverwaltung
- Zuverlässigkeit, Sicherheit und Qualität besonders für den kommerziellen Einsatz
- Produkte für den Rechenzentrumseinsatz, die auf Grund jahrelanger Erfahrung den Bedürfnissen und Erwartungen der kommerziellen Kunden entgegenkommen
- stabile kommerzielle Anwendungen für den Einsatz auf Großrechnern, wie z.B. Datenbanksysteme oder Transaktionsmonitore

SINIX-Anwendungen

Siemens Nixdorf vertreibt und unterstützt eine breite Palette von Anwendungen für SINIX-Systeme. Diese reichen von allgemeinen Büroanwendungen bis zu vertikalen Marktlösungen, wie sie von Siemens Nixdorf z.B. für das Bankwesen, den Einzelhandel und für das Gastronomie- und Hotelgewerbe angeboten werden. Produktinformationen und Informationen darüber, wie Sie diese Produkte bestellen können erhalten Sie von den Siemens Nixdorf Zweigniederlassungen.

Daneben gibt es unzählige Softwarehäuser, die ihre Anwendungen speziell für SINIX-Systeme anbieten. Bei Siemens Nixdorf wurden Hunderte dieser Angebote gesammelt und in der *SINIX Softwarebörse* (englisch: *SINIX Software Exchange*) veröffentlicht.

Eine Fülle von Anwendungen wurde für alle Varianten von UNIX entwickelt und auf den Markt gebracht und kann damit auch auf SINIX-Systemen eingesetzt werden. Diese sind im *UNIX-Softwareführer aufgelistet* (englisch: *UNIX Software Guide*), zu beziehen über die Siemens Nixdorf Zweigniederlassungen.

Aufbau des Buchs

Das zweite Kapitel ab Seite 9 beschäftigt sich mit der Entwicklung von SINIX aus dem Betriebssystem UNIX. Es beschreibt die Besonderheiten von UNIX, die dazu führen, daß UNIX an erster Stelle genannt werden muß, wenn man von offenen Systemen spricht. Dazu gehören insbesondere die Portabilität von UNIX und der für UNIX entwickelten Anwendungen.

Das dritte Kapitel ab Seite 21 beschäftigt sich mit Kommunikation und der Netzwerkfähigkeit von SINIX. Die andere hervorragende Eigenschaft eines offenen Systems neben der Portabilität ist die Fähigkeit, mit anderen Systemen im Netzverbund zu kommunizieren und zusammenzuarbeiten.

Die Kapitel vier, fünf und sechs ab Seite 39 befassen sich mit den Arbeitsumgebungen, die SINIX für die drei wichtigsten Anwendergruppen eines Computersystems bereithält: nicht-privilegierte Anwender, Programmentwickler, System- und Netzverwalter.

Kapitel sieben ab Seite 123 beschreibt die verteilte Verarbeitung. Dies ist ein relativ neuer, aber zukunftsträchtiger Ansatz, sich die Stärken von Computernetzen nutzbar zu machen.

Kapitel acht ab Seite 151 beschreibt detailliert, wie SINIX die Bedürfnisse der kommerziellen Datenverarbeitung und großer Rechenzentren befriedigen kann.

Kapitel neun ab Seite 177 beschreibt, warum SINIX durch seine besonderen Leistungsmerkmale im Bereich Verfügbarkeit, Sicherheit und Qualität für den Einsatz im kommerziellen Bereich besonders geeignet ist.

Kapitel zehn ab Seite 209 wagt einen Ausblick in die Zukunft und weist auf einige technische Neuentwicklungen hin, die großen Einfluß auf den kommerziellen Einsatz von Computersystemen zu nehmen beginnen.

Den Abschluß bilden die Anhänge. Anhang A ab Seite 241 ist für Leser gedacht, die sich besonders für die Standards von offenen Systemen interessieren. Neben verschiedenen Standardisierungsgremien und Industriekonsortien werden auch die Standards beschrieben, die von diesen Organisationen und Konsortien entwickelt wurden. Anhang B ab Seite 253 enthält ein Verzeichnis wichtiger Abkürzungen, die in diesem Buch Verwendung finden sowie deren Auflösungen. Anhang C ab Seite 259 enthält ein Verzeichnis der verwendeten Literatur.

2 Das Betriebssystem SINIX

Überblick

Wenn ein Unternehmen einen Rechner anschafft, spielt die Frage nach dem Betriebssystem, das auf diesem Rechner läuft, eine wichtige Rolle. Die Auswahl eines Betriebssystems hat Auswirkungen auf die drei unterschiedlichen Gruppen, die mit dem Rechner arbeiten werden: die Anwender, die Anwendungsentwickler und die System- oder Netzverwalter.

Noch vor wenigen Jahren war die Schnittstelle zwischen Rechner und Anwender die Kommandozeile, über die Kommandos eingegeben wurden, die eine Anwendung starteten, den Inhalt einer Festplatte ausgaben, eine Datei anzeigten, usw. Dabei machte es keinen Unterschied, ob es sich bei dem Rechner um einen MS-DOS-basierten PC, ein UNIX-basiertes Midrange-System oder auch einen Mainframe-Rechner handelte. Heute jedoch stellen die Betriebssysteme mit ihren grafischen Bedienoberflächen dem Anwender eine benutzerfreundlichere und natürliche Interaktionsmöglichkeit mit dem Rechner zur Verfügung. Außerdem wird damit die Möglichkeit geschaffen, Anwendungen zu erstellen, die die Vorteile der grafischen Bedienoberflächen ausnutzen.

Für den Anwendungsentwickler war das Betriebssystem schon immer eine entscheidende Komponente seiner Arbeitsumgebung. Das Betriebssystem stellt die Bibliotheksroutinen, Systemaufrufe und andere Mechanismen zur Verfügung, die der Entwickler für die Programmierung seiner Anwendungen verwenden kann. In einem umfassenderen Sinn ist das Betriebssystem für den Entwickler wichtig, weil es zudem bestimmt, welche Werkzeuge zusätzlich zur Verfügung stehen: Compiler für High-Level Programmiersprachen wie C und COBOL, Editoren, Debugger, usw. Der Anwendungsentwickler muß normalerweise einen großen Teil seiner Arbeit auf eine gut ausgedachte, bedienungsfreundliche Bedienoberfläche verwenden. Auch

hier haben die Betriebssystemumgebung und die Programmierwerkzeuge einen großen Einfluß darauf, ob man als Entwickler, schnell und effektiv eine gute Bedienoberfläche erstellen kann.

Für Anwendungen, die eine Kommunikation über ein Netzwerk beinhalten, muß sich der Anwendungsentwickler mit Einrichtungen des Betriebssystems oder verfügbaren Zusatzprodukten befassen, die die Kommunikation mit anderen Rechnern über ein Netzwerk unterstützen.

Aus der Sicht eines System- oder Netzverwalters stellt das Betriebssystem Komponenten für administrative Aufgaben wie das Einrichten von Benutzerkennungen, die Datensicherung, die Sicherheitsvorkehrungen, die Unterstützung von Netzverbindungen, usw. zur Verfügung. Die System- oder Netzverwalter verwenden oft eine ganze Reihe von Betriebssystemschnittstellen und Dienstprogrammen, die von Anwendungsentwicklern selten und noch weniger von Anwendern benutzt werden. Die Sicht eines Verwalters auf das Betriebssystems ist in der Regel geprägt von der Verfügbarkeit und Effektivität dieser Schnittstellen und Dienstprogramme.

Aus all dem wird ersichtlich, daß die Entscheidung eines Unternehmens, welche Rechner angeschafft und mit welchem Betriebssystem sie betrieben werden, sehr komplex ist. Die Entscheidung muß die unterschiedlichen Bedürfnisse und Anliegen der verschiedenen Unternehmensteile berücksichtigen. Wir glauben, daß heute in den meisten Fällen ein standardisiertes UNIX-Betriebssystem wie SINIX die beste Wahl für alle Unternehmensteile und das Unternehmen insgesamt darstellt.

UNIX: Ein offenes Betriebssystem

Das Betriebssystem UNIX wurde ursprünglich in den frühen siebziger Jahren von Ken Thompson und Dennis Ritchie in den Entwicklungslabors von AT&T als Werkzeug für den internen Gebrauch entwickelt. Aus diesem Anfang entwickelte es sich von einem Programmierwerkzeug im Forschungslabor zu dem heutigen kommerziell nutzbaren und erfolgreichen Betriebssystem.

Aufgrund folgender Stärken wurde UNIX dabei zu dem Betriebssystem für offene Systeme:

- Portabilität des Betriebssystems bezüglich unterschiedlicher Hardwareplattformen
- Portabilität der UNIX-Anwendungsprogramme
- „Skalierbarkeit", d.h. Einsatzmöglichkeit vom kleinen bis zum großen Rechner
- leistungsstarke Software-Entwicklungsumgebung
- hervorragende Dienstprogramme
- Unterstützung des Multitasking- und Multiuser-Betriebs
- überragende Unterstützung für die Integration in Netzwerken

Die Architektur von UNIX ist durch eine hohe Portabilität und Skalierbarkeit geprägt. Sie kann leicht für viele verschiedene Rechner angepaßt werden und eignet sich für einen großen Leistungsbereich, vom High-end-PC bis hin zum Mainframe-Rechner. UNIX ist klar strukturiert. Es gibt einen kleinen Systemkern, der alle Hardwareabhängigkeiten behandelt und die System- und Anwendungsprogramme, die auf dem Systemkern aufsetzen. Alle hardwareabhängigen Teile sind im Systemkern untergebracht, aber nur 10% des Systemkerns sind hardwareabhängig.

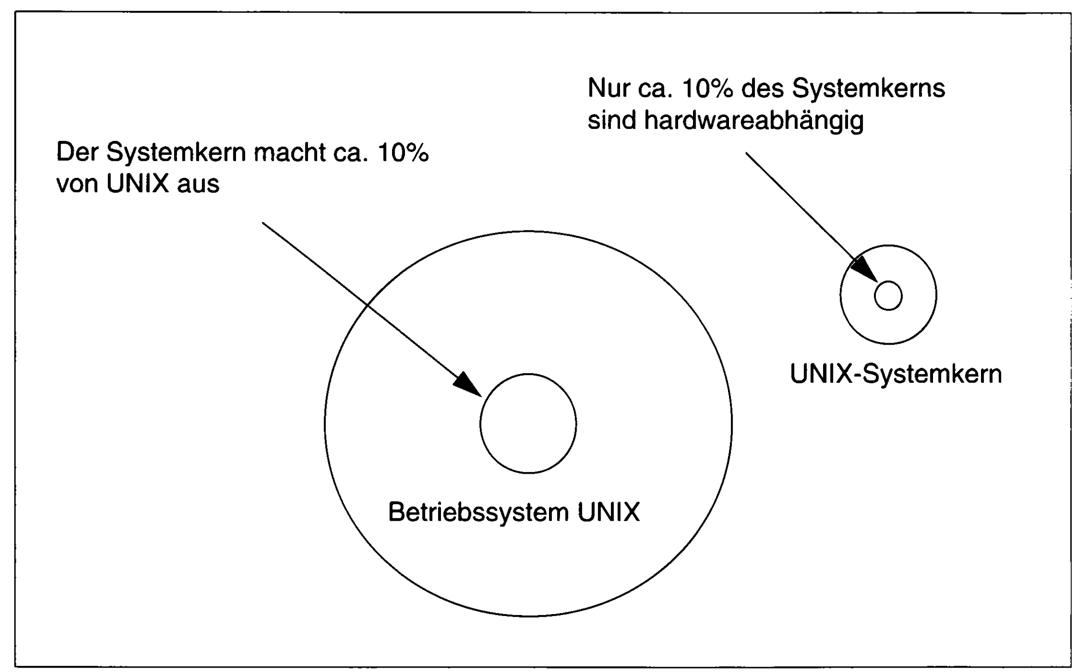


Abbildung 2-1: UNIX zeichnet sich durch seine Portabilität aus. Nur ungefähr 1% des Betriebssystems ist hardwareabhängig.

Bis auf sehr kleine Teile ist UNIX in der High-Level Programmiersprache C geschrieben. Dies vereinfacht die Portierung von UNIX auf weitere Plattformen. UNIX kann mit einem geringen Aufwand auf neue Hardwarearchitekturen angepaßt, sprich portiert werden. Dies ist der Grund dafür, daß UNIX heute hinsichtlich der Anzahl der Rechnerarchitekturen, auf denen es betrieben werden kann, an der Weltspitze liegt.

Wollte man den Grund für den Erfolg von UNIX mit einem einzigen Wort beschreiben, könnte man das Wort „Portabilität" verwenden. Die Portabilität von UNIX hat dazu geführt, daß UNIX von vielen verschiedenen Anbietern propagiert wird, ein universelles Anwendungsspektrum und eine hohe Akzeptanz bei Entwicklern und Anwendern besitzt. Hand in Hand damit ging die „Offenheit“ von UNIX, d.h. die allgemeine Verfügbarkeit der Schnittstellen und der zugehörigen Betriebssystem-Software. Die beiden Worte „Offenheit“ und „Portabilität“ beschreiben wohl am besten die Idee, die anfangs hinter UNIX stand. Heute muß man in diesem Zusammenhang ebenso die Interoperabilität hervorheben.

Die UNIX-Entwicklungslinien

In der Vergangenheit gab es drei UNIX-Hauptentwicklungslinien:

- **BSD UNIX** (**B**erkeley **S**oftware **D**istribution) fand hauptsächlich im technisch-wissenschaftlichen Bereich und auf Workstations Verbreitung. Sun Microsystems wählte BSD als Basis für ihr Betriebssystem SunOS und fügte als wesentliche Neuerung das NFS (**N**etwork **F**ile **S**ystem) hinzu.
- **UNIX System V** ist das von AT&T weiterentwickelte UNIX. System V konnte sich als „Standard UNIX" im kommerziellen Bereich bei Mehrplatzsystemen durchsetzen.
- **XENIX** deckte den „Low-end"-Bereich ab und wurde hauptsächlich aufgrund des geringen Verbrauchs an Ressourcen auf PCs eingesetzt.

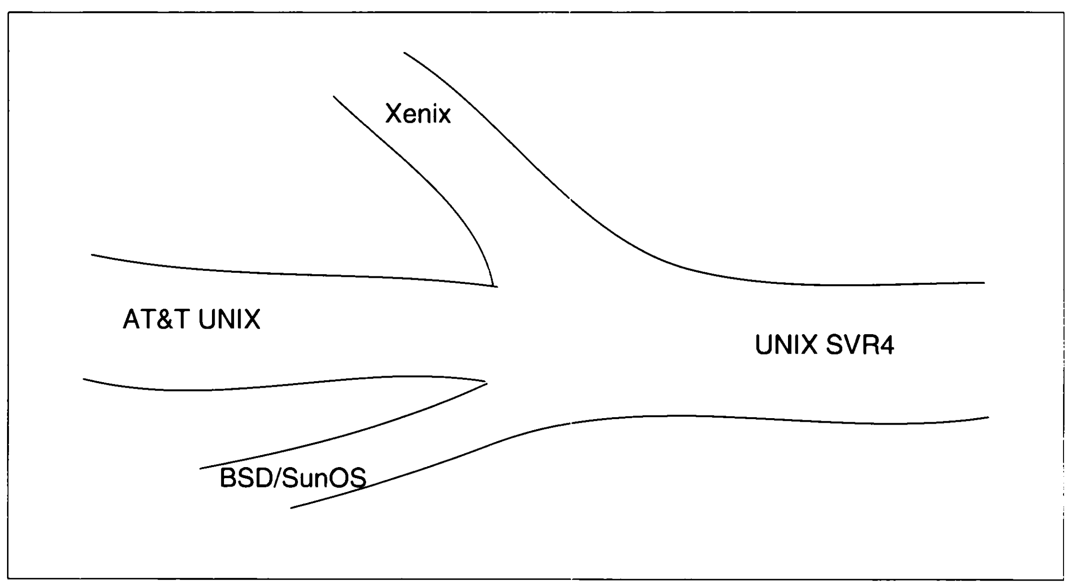


Abbildung 2-2: Drei UNIX-Entwicklungslinien sind in UNIX SVR4 zusammengeführt worden.

UNIX System V Release 4

UNIX System V Release 4 (SVR4) ist das Ergebnis der Bemühungen, die Divergenzen von UNIX-Varianten zu beenden. SVR4 vereint die Merkmale der drei Hauptentwicklungslinien in einer einzigen UNIX-Version. Hier einige der herausragenden Merkmale von SVR4:

- Durchsetzung als Industriestandard
- Spezifizierung eines Binärstandards (**A**pplication **B**inary **I**nterface)
- Verwendung gemeinsam benutzter Bibliotheken
- Echtzeitverarbeitung
- Hauptspeicherverwaltung
- Prozeßverfolgung
- virtuelles Dateisystem

Industriestandards

Industriestandards ermöglichen es den Software-Anbietern, Anwendungen zu erstellen, die einfach zu portieren sind und deren Wert damit auch in Zukunft erhalten bleibt. In der Vergangenheit wurden, anders als bei Siemens Nixdorf, von vielen Anbietern eigene UNIX-Versionen vermarktet und unterstützt. Sie blieben nicht beim erstmals von AT&T und später von UNIX System Laboratories (USL) bereitgestellten Standard-UNIX. Diese anbieter-spezifischen Versionen von UNIX waren eine Mischung aus einem offenen System, da es sich weitgehend um ein UNIX-Derivat handelte, und einem proprietären System, da charakteristische Modifikationen der Anbieter eingebaut waren.

Es gibt aber immer mehr Anzeichen dafür, daß diese Firmen sich allmählich in Richtung auf das Standard-UNIX (das System V Release 4 der UNIX System Laboratories) bewegen. Zum einem, um die eigenen Kosten zu reduzieren und zum anderen, um den Kunden ein offeneres System anbieten zu können.

Binärstandard

Die Einhaltung von Standards sichert den Softwareentwicklungsfirmen die Portierbarkeit von Software auf der Quellprogramm-Ebene. Der Käufer einer Software bekommt diese aber normalerweise in binärer Form. In der PC-Welt mit ihren Klones ergeben sich daraus keine Probleme. Die meiste „von der Stange“ gekaufte Software ist auf allen Rechnern ablauffähig, da alle Rechner der gleichen Prozessorfamilie angehören. Da das Betriebssystem UNIX auf einer ganzen Reihe von Prozessortypen ablauffähig ist, wäre eine einzige Binärform für UNIX nicht denkbar. Es ist aber möglich, die gleiche UNIX-Software in binärer Form auf Rechnern des gleichen Prozessortyps verschiedener Anbieter ablaufen zu lassen. Sowohl Hersteller als auch Käufer der Software profitieren von dieser Binärcodekompatibilität.

UNIX SVR4 besitzt einen solchen Binärstandard, das UNIX-ABI (**A**pplications **Bi**nary **I**nterface). Es handelt sich hierbei um eine detaillierte Beschreibung von System- und Compilerschnittstellen. Diese Spezifikation ist in einen generischen und einen prozessorspezifischen Teil untergliedert. Es existieren gültige ABI-Spezifikationen für die Prozessorarchitekturen von Intel x86, Motorola, SPARC und MIPS. Das ABI garantiert die binäre Ablauffähigkeit konformer Anwenderprogramme. Ein zukünftiges Ziel ist es, noch einen DDI/DKI-Standard (**D**evice **D**river **I**nterface, **D**river-**K**ernel **I**nterface) zu etablieren.

Gemeinsam benutzte Bibliotheken

Der Mechanismus der gemeinsam benutzten Bibliothek „dynamic shared library" erlaubt es, daß mehrere Anwendungen sich den gleichen Code aus einer Bibliothek teilen. Die Bibliothek ist also nicht mehr fester Bestandteil der Anwendung, sondern wird nur noch einmal, eventuell für mehrere Prozesse, in den Hauptspeicher geladen. Dadurch reduziert sich die Größe des einzelnen Objektprogramms und spart somit Festplattenkapazität, zum anderen wird aber auch weniger Hauptspeicher für mehrere geladene Prozesse benötigt. Dies wird durch einen dynamischen Binder ermöglicht, der erst zur Laufzeit, beim Zugriff des Prozesses auf die Bibliothek, die Adressen der Bibliotheksroutinen berechnet. Neben der Einsparung an Ressourcen besteht der Hauptvorteil darin, Bibliotheken austauschen zu können, ohne die Anwendung erneut binden zu müssen. Das führt zu enormen Kostenreduzierungen bei der Auslieferung und Installation von Fehlerkorrekturen.

Echtzeitverarbeitung

Ursprünglich wurde UNIX als Time-Sharing-System konzipiert. Dies bedeutet, daß sich alle lauffähigen Prozesse um die CPU bewerben. Das Ziel war, die CPU-Zeit gleichmäßig auf alle Prozesse zu verteilen, eine für den Mehrbenutzer-Betrieb notwendige Voraussetzung. Mit der zunehmenden Verbreitung von UNIX entstand auch der Wunsch nach einem Einsatz im Bereich der Fertigungssteuerung. Das erfordert, daß gewisse Prioritäten vergeben werden können, so daß ein Antwortverhalten in Echtzeit garantiert werden kann. Um die notwendigen Echtzeitanforderungen zu befriedigen, wurden in SVR4 die zwei Auftragsklassen „Time-Sharing“ und „Real-Time“ eingeführt. Der Systemverwalter hat damit die Möglichkeit, die Zeitscheibe, die ein Prozeß erhält und dessen Priorität festzulegen.

Hauptspeicherverwaltung

Die Hauptspeicherverwaltung unter SVR4 wurde weitgehend von SunOS übernommen. Wie bei den UNIX-Varianten AT&T UNIX und BSD wurde ein „Demand-Paging“-Konzept realisiert, d.h. nur diejenigen Seiten eines Prozesses werden in den Hauptspeicher geladen, die aktuell benötigt werden. Nicht benötigte Seiten können bei Speicherengpässen wieder auf die Festplatte ausgelagert werden. Völlig neu dagegen ist die Art der Realisierung in SVR4. Als grundlegendes Konzept dienen sog. „Mapped Files“, d.h. beliebige Dateien können im virtuellen Adreßraum eines Prozesses abgebildet werden. Um einen Prozeß zu starten, wird das ausführbare Programm auf der Festplatte vom Systemkern abgebildet. Der Vorteil der neuen Architektur der virtuellen Speicherverwaltung besteht in einer effizienteren Ausnutzung des Hauptspeichers, da sich mehrere Prozesse die gleichen Seiten teilen können. Da es außerdem möglich ist, physikalische Geräte in den Prozeßraum zu legen, ergeben sich elegante Möglichkeiten, auch auf die Hardware zuzugreifen. Beim Entwurf der neuen virtuellen Speicherverwaltung wurde streng darauf geachtet, eine Unterteilung in hardwareabhängige und hardwareunabhängige Teile vorzunehmen. Die Portierung des Systems auf eine neue Hardwareplattform verursacht deshalb geringere Aufwände im Vergleich zu den alten UNIX-Varianten.

Prozeßverfolgung

Das */proc*-Dateisystem ermöglicht es, auf jeden geladenen Prozeß zuzugreifen und Diagnose- und Statusinformationen zu erhalten. Dies erleichtert die Fehlersuche in Programmen, vor allem wenn die Sourcen einer Anwendung nicht verfügbar sind.

Virtuelles Dateisystem

Um alle vorhandenen UNIX-Dateisysteme zu integrieren war es notwendig, zwischen generischen und Dateisystem-spezifischen Teilen zu unterscheiden. Das virtuelle Dateisystemkonzept des VFS (**V**irtual **F**ile **S**ystem) bietet sowohl eine einheitliche Benutzerschnittstelle für alle Dateisystemtypen, als auch eine generische Schnittstelle im Systemkern, damit neue Dateisysteme dem UNIX-Systemkern hinzugefügt werden können.

Die folgenden Dateisysteme sind in SVR4 integriert:

- UFS: Berkeley Fast File System
- S5: Original AT&T UNIX Dateisystem
- NFS: Network File System der Firma Sun
- RFS: Remote File Sharing

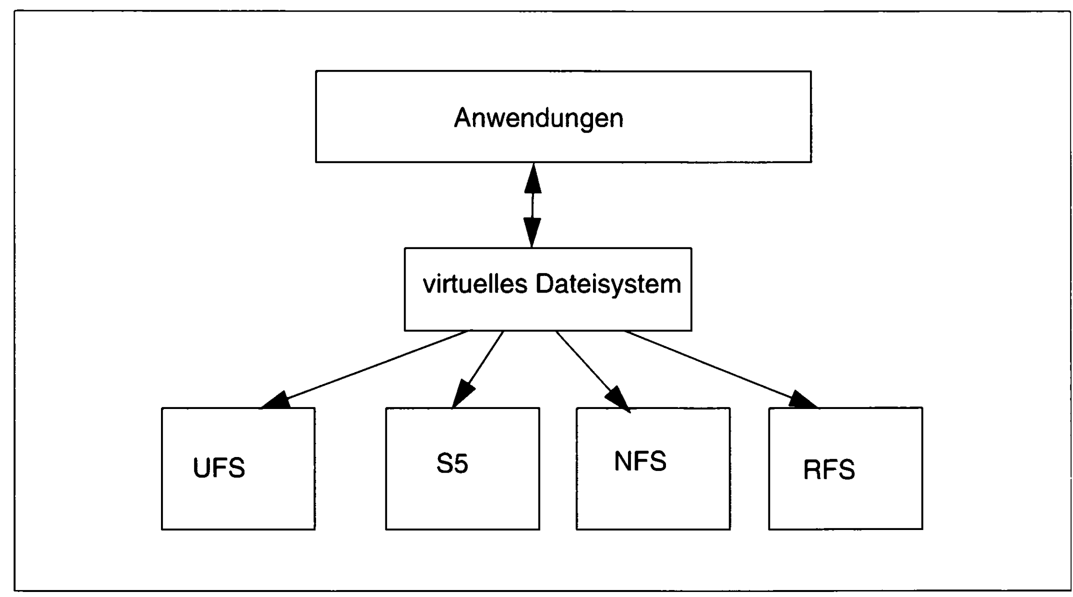


Abbildung 2-3: Das virtuelle Dateisystem von SVR4 stellt allen Anwendungen eine einheitliche Schnittstelle für verschiedene Dateisysteme zur Verfügung.

SINIX

SINIX basiert auf System V Release 4. SINIX ist UNIX auf einer leistungsfähigen, stabilen und modular konfigurierbaren Hardware, auf PCs, auf Workstations, Mehrbenutzer- und Serversystemen. Aber SINIX ist mehr als nur ein UNIX. Es ist ein UNIX mit „Mehrwert".

Offene Kommunikation

Das Betriebssystem SINIX stellt alle wesentlichen Kommunikationsschnittstellen und -protokolle zum Aufbau von Hochleistungs-Netzwerken mit anderen UNIX-Systemen, PCs, Workstations und Mainframe-Rechnern zur Verfügung (siehe Kapitel "3 Kommunikation und Netzwerke" auf Seite 21).

Anwenderfreundliche Bedienoberflächen

Schon die frühen SINIX-Systeme besaßen ein leicht verständliches Menü-Bediensystem für die Verwaltung und Benutzung des Rechners sowie das Starten von Programmen. Diese Tradition wurde fortgesetzt, als mit dem Fenstersystem Collage eine grafische Benutzerschnittstelle auf Einbenutzer- und Mehrbenutzer-Systemen implementiert wurde. Heute werden die grafischen Benutzerschnittstellen in SINIX mit dem X Window System (entwickelt am Massachusetts Institute of Technology und standardisiert im X/OPEN XPG3 und XPG4) und OSF/Motif erstellt (siehe Abschnitt „Das X Window System" auf Seite 41).

Breites Software-Angebot

SINIX selbst wurde im Lauf der Zeit durch viele bedeutende und wünschenswerte Funktionen immer weiter verbessert. So wurden zum Beispiel die Sicherheit und die Verfügbarkeit erhöht. SINIX bietet außerdem ein leistungsfähiges Spoolsystem, ein verteiltes Dateisystem (NFS), ein Dateisystem, das die Verfügbarkeit erhöht (VxFS), und vieles mehr. Viele Softwareprodukte werden zusätzlich zur Verwaltung und Bedienung von SINIX-Rechnern angeboten. Zur Entwicklung von Anwendungsprogrammen stehen verschiedene Compiler (C, C++, COBOL, FORTRAN, BASIC, PASCAL-XT, ADA, PROLOG) und verschiedene Entwicklungsumgebungen (Editoren und Debugger) zur Verfügung. Verteilte Anwendungen können mit Hilfe von Datenbanksystemen, einem Transaktionsmonitor und dem „**D**istributed **C**omputing **E**nvironment" (DCE der Open Software Foundation) implementiert werden. Diese und auch andere Produkte werden in den folgenden Kapiteln beschrieben.

Breites Hardware-Angebot — Vom Schreibtisch bis zum Rechenzentrum

Das Betriebssystem UNIX ist das einzige Betriebssystem, das nahezu die gesamte Palette der heute erhältlichen Rechner abdeckt. Die Palette reicht vom Monoprozessorsystem über Multiprozessorsysteme, vom einfachen PC bis in den Mainframe-Bereich, in dem verschiedene Prozessortechnologien verwendet werden.

Seit dem ersten SINIX-System, das mit einem Intel 8086 Prozessor arbeitete, hat Siemens Nixdorf SINIX auf jede neue Prozessorgeneration portiert, um den Kunden die beste Hardwaretechnologie und das beste Preis-/Leistungsverhältnis zu bie-

ten. Siemens Nixdorf bietet SINIX auf einer Reihe von Plattformen an, vom PC über Workstations zu großen Mehrplatzsystemen. SINIX-Systeme mit der gleichen Prozessorarchitektur sind binärkompatibel; für unterschiedliche Prozessorarchitekturen gilt eine volle Sourcekompatibilität auf der Anwendungsebene. Die Schnittstellen sind im SINIX API (**A**pplication **P**rogramming **I**nterface) definiert (siehe Abschnitt „SINIX API" auf Seite 75).

Der Vorteil eines einheitlichen Betriebssystems für die gesamte Palette an Hardwareplattformen ist offensichtlich. Anwendungen, die auf einer Plattform entwickelt wurden, sind auf allen Rechnern ablauffähig. Handelt es sich um unterschiedliche Prozessorarchitekturen, muß nur neu übersetzt werden. Die Arbeitsumgebung für den Anwender und die Mittel zur Systemverwaltung sind auf allen Rechnern die gleichen, wodurch die Kosten für Ausbildung und Verwaltung reduziert werden.

SINIX läuft auf PCs, auf CISC (**C**omplex **I**nstruction **S**et **C**omputer) Midrange-Rechnern und auf RISC (**R**educed **I**nstruction **S**et **C**omputer) Midrange- und High-End-Rechnern.

3 Kommunikation und Netzwerke

Überblick

Schon seit langem bieten Computerhersteller proprietäre Lösungen für Kommunikation und Netzwerke an. Im allgemeinen beschränken sich diese Lösungen jedoch auf homogene Kommunikationssysteme, die auf die Produkte des jeweiligen Herstellers zugeschnitten sind. Der Funktionsumfang der Netzwerksysteme wurde später im Rahmen von herstellerunabhängigen, übergreifenden Lösungen erweitert, so z.B. um elektronische Post (E-Mail), Anmelden an ferne Systeme oder Dateitransfer. Schließlich kamen Produkte mit dem Versprechen auf den Markt, auf ganz unterschiedlichen DV-Systemen den vollen Leistungsumfang für eine offene Kommunikation anzubieten.

Dieses Kapitel beschäftigt sich zuerst mit einigen Netzwerkprotokollen, die aus solchen herstellerspezifischen Produkten hervorgingen und heute als De-facto-Standard anerkannt werden. Im Anschluß daran werden zwei Netzarchitekturen vorgestellt, nämlich die Internet-Protokoll-Familie und die OSI-Protokoll-Familie. Darauf folgt eine Einführung, wie PCs in Netzwerke integriert werden. Vor diesem Hintergrund werden schließlich wichtige Kommunikations- und Netzwerkprodukte von Siemens Nixdorf vorgestellt.

De-facto-Standardprotokolle

Eine Reihe von Computerherstellern hatte schon eigene Kommunikationsprotokolle entwickelt, bevor standardisierte Kommunikationsprotokolle auf den Markt gebracht wurden. Dazu gehörten die System Network Architecture von IBM (SNA), die Digital Network Architecture der Digital Equipment Corporation und die Network Environment Architecture (NEA) von Siemens Nixdorf. Diese Protokolle waren in erster Linie für den Einsatz in einem homogenen Computernetz gedacht, einen Netzverbund also, der nur aus Rechnern eines Herstellers besteht. Es ist natürlich möglich, Rechner unterschiedlicher Hersteller miteinander zu verbinden, dafür ist jedoch meist eine aufwendige Umsetzung der Protokolle nötig, selbst dann, wenn die passenden Angaben von der Firma zur Verfügung gestellt werden, die die Rechte an diesem Protokoll besitzt (Siemens Nixdorf hat dies so gehandhabt). Zur Durchführung einer solchen Umsetzung ist häufig der Einsatz zusätzlicher Hardware erforderlich.

NEA steht für eine Reihe von Siemens-Nixdorf-spezifischen Protokollen, die von Siemens Nixdorf zu einer Zeit entwickelt wurden, als Protokolle mit vergleichbarem Standard von anderer Seite noch nicht verfügbar waren. Siemens Nixdorf ist den offenen Kommunikationsprotokollen verpflichtet und unterstützt diese auch in den eigenen Produkten. Um unsere Kunden optimal zu versorgen, werden wir aber auch weiterhin solche Produkte pflegen, die auf den NEA-Protokollen basieren. So arbeiten z.B. die Siemens-Nixdorf-Produkte der TRANSDATA-Serie NEA-konform. Mit der Zeit wurde die Palette der TRANSDATA-Produkte erweitert. Sie unterstützen nun eine breite Palette von Protokollen, die entweder einen De-facto-Standard darstellen oder von offizieller Seite als Standard anerkannt werden, so z.B. SNA, TCP/IP, OSI oder ISDN.

Internet-Protokollfamilie

Die ursprüngliche Arbeit an der Internet Protokollfamilie wurde hauptsächlich von einer Dienststelle des amerikanischen Verteidigungsministeriums, der **D**efense **A**dvanced **R**esearch **P**rojects **A**gency (DARPA), finanziert. Von der DARPA wurde auch das erste weltumspannende Netzwerk mit dieser Technologie, das Internet, ins Leben gerufen. Das Wort internet (klein geschrieben) wird verwendet, wenn man allgemein von einem Netzwerk spricht, das auf der Internet-Technologie beruht, also mehrere Netzwerke untereinander verbindet. Das Wort Internet (groß geschrieben) wird immer dann verwendet, wenn genau dieses eine Netzwerk der DARPA gemeint ist.

Die Internet-Protokoll-Familie entwickelte sich aus einem Forschungsauftrag für ein Paketnetzwerk. Dieses sollte unterschiedliche Übertragungsmedien unterstützen und sogar in einer Katastrophensituation sicher funktionieren. Am Anfang gab es nur ein Netzwerk, ARPANET, das ein paar Dutzend Computersysteme in den USA miteinander vernetzte. Mit dem Aufkommen von verschiedenen Netzwerktechnologien, wie z.B. Ethernet, Packet Radio oder Satellitenübertragung, wurde eine Methode für die zuverlässige Informationsübertragung über Medien benötigt, die keine zuverlässige und fehlerfreie Übertragung garantieren. Informationen, die mit Hilfe dieser Medien übermittelt werden, können z.B. durch Radiostörungen oder Kollisionen verstümmelt werden oder ganz verloren gehen.

Die Internet-Protokollfamilie basiert im wesentlichen auf zwei Diensten:

- einem verbindungorientierten Transportdienst, der vom **T**ransmission **C**ontrol **P**rotocol (TCP) zur Verfügung gestellt wird
- einem verbindungslosen Netzwerkdienst, der vom **I**nternet **P**rotocol (IP) zur Verfügung gestellt wird

Das Internet-Protokoll bietet auf der Transportebene zwei Dienste an. Das ist zum einen das **U**ser **D**atagram **P**rotocol (UDP). Es handelt sich dabei um ein verbindungsloses Transportprotokoll der Internet-Familie. Da UDP ein Protokoll der Transportschicht ist, benutzt es für die Zustellung IP. Zum anderen ist dies das **T**ransmission **C**ontrol **P**rotocol (TCP). Es handelt sich dabei um ein verbindungsorientiertes Transportprotokoll aus der Internet-Familie. Da TCP ein verbindungsorientiertes Transportprotokoll ist, durchläuft es drei verschiedene Phasen: Aufbau der Verbindung, Datenübertragung und schließlich Abbau der Verbindung.

Internet-Dienste

Es gibt mehrere Anwendungsprotokolle in der Internet-Familie, die ständig benutzt werden:

- Das **F**ile **T**ransfer **P**rotocol (FTP), welches Dienste für die Dateiübertragung zur Verfügung stellt. Im Vergleich zum FTNEA-Protokoll von Siemens Nixdorf oder auch zum ISO-Standard-Protokoll FTAM ist der Funktionsumfang des FTP jedoch gering.
- TELNET, das ein virtuelles Terminal zur Verfügung stellt.
- Das **S**imple **M**ail **T**ransfer **P**rotocol (SMTP), welches einen Speicher- und Weiterleitungsdienst für elektronisch übermittelte Nachrichten zur Verfügung stellt.
- Der **D**omain **N**ame **S**ervice (DNS), welcher hauptsächlich eine Abbildung von Rechnernamen auf Netzwerkadressen anbietet.

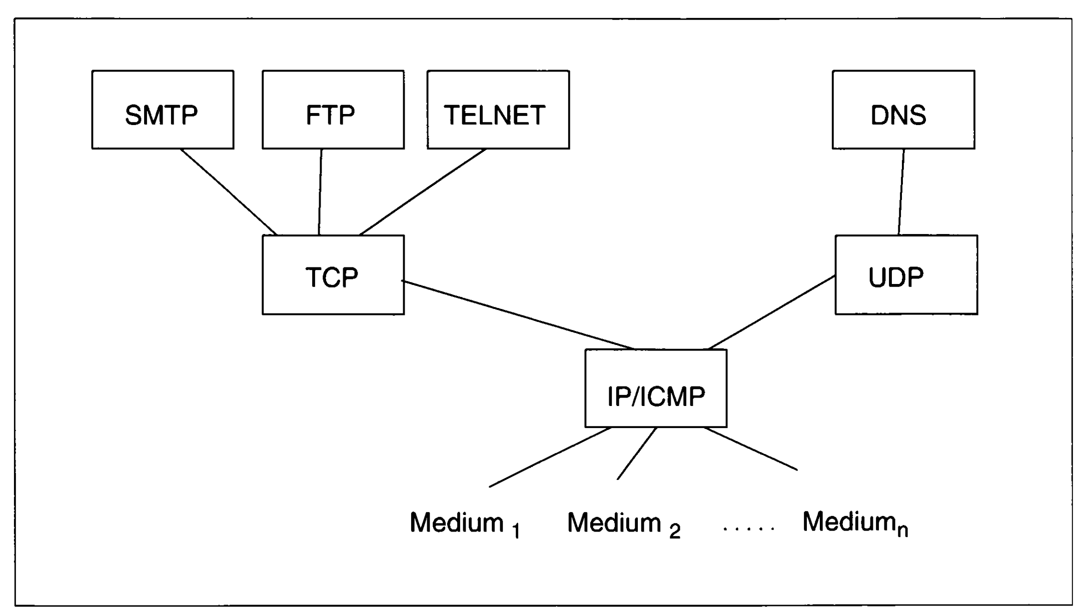


Bild 3-1. Aufbau der Internet-Dienste

Abbildung 3-1 zeigt anhand einer einfachen Übersicht die Internet-Dienste und die dabei verwendeten Protokolle. Beim ICMP, einem Protokoll, auf das in der Abbildung Bezug genommen wird, handelt es sich um das Internet Control Message Protocol. Es wird verwendet, um die Internetschicht zu überwachen und hängt eng mit IP zusammen.

Es gibt natürlich weitaus mehr Dienste, die auf TCP/IP aufsetzen und so unterschiedliche Einsatzgebiete haben wie Netzwerkmanagement oder Sprachprotokolle. Die beiden Dienste mit der größten Bedeutung sind derzeit

- NFS, ein verteiltes Dateisystem, das von Sun Microsystems entwickelt wurde. Das NFS wird später genauer beschrieben (siehe Abschnitt „Das verteilte Dateisystem NFS" auf Seite 127).
- das X Window System, das am Massachusetts Institute of Technology entwickelt wurde. Das X Window System wird später näher beschrieben (siehe Abschnitt „Das X Window System" auf Seite 41).

Es ist bemerkenswert, daß beide Anwendungsprotokolle so aufgebaut sind, daß sie von den tatsächlichen Ende-zu-Ende-Diensten zur Datenübertragung unabhängig bleiben. Dennoch werden beide, sowohl aus technischen als auch aus wirtschaftlichen Gründen, am häufigsten in Verbindung mit den Protokollen der Internet Protokollfamilien eingesetzt.

OSI-Schichtenmodell

Im Jahr 1977 begann die **I**nternational **S**tandards **O**rganization (ISO) mit der Arbeit an einem Standard-Schichtenmodell für Computerkommunikation. Man entwickelte das sogenannte Open-Systems-Interconnection-Schichtenmodell. Das OSI-Schichtenmodell hat sieben Protokollschichten. Die folgende Übersicht gibt die Protokollschichten an und zeigt, welche Aufgaben diese im einzelnen übernehmen:

Schicht 1: Physikalische Ebene	
	Kontrolle der physikalischen Verbindung
	Bitübertragungsprotokoll
Schicht 2: Verbindungsebene	
	Auf- und Abbau der Verbindungen zwischen Systemen
	Sicherungsprotokoll
Schicht 3: Netzwerkebene	
	Routing
	Vermittlungsprotokoll
Schicht 4: Transportebene	
	Aufbau von Transportverbindungen
	Integrität und Multiplexen der Datenübertragung
Schicht 5: Sitzungsebene	
	Zuweisen von Sitzungen an Transportverbindungen
	Überwachen des Datenaustauschs durch Kontrollmechanismen
Schicht 6: Präsentationsebene	
	Anfordern einer Sitzung
	Koordination der Syntax
	Festlegen des Datenformats
Schicht 7: Anwendungsebene	
	Kontrolle der Anwendungen
	Kommunikationsmanagement zwischen den Anwendungen

Das OSI-Schichtenmodell ist nicht an bestimmte Anwendungen gebunden. Der Sinn dieser Architektur liegt in der hierarchischen Strukturierung der Bedürfnisse eines Kommunikationssystems, damit Softwarehäuser für jede dieser Strukturebenen Produkte anbieten können, die dann ohne Schwierigkeiten in umfassende Lösungen eingefügt werden können. Bei diesem Modell ist von entscheidender Bedeutung, daß es genau festgelegte Schnittstellen zwischen den einzelnen Schichten gibt. Denn es geht davon aus, daß aufgrund einer stillschweigenden Übereinkunft jede Schicht ihre Dienste der darüberliegenden Schicht an einer geeigneten Schnittstelle zur Verfügung stellt.

Die sieben Schichten kann man in Transportdienste (Ebenen 1 bis 4) und Anwendungsdienste (Ebenen 5 bis 7) unterteilen. Die Transportdienste stellen einen Ende-zu-Ende-Dienst zur Verfügung, der für die Datenübertragung verantwortlich ist. Dies wird in einem Buch über das OSI-Modell näher erläutert: „Ende-zu-Ende-Dienste kümmern sich um die Datenübertragung. Während sich die Anwendungsdienste um die Informationsvermittlung kümmern, übertragen die Ende-zu-Ende Dienste nur Oktette (8-bit-Werte) von einem System zu einem anderen. Dabei spielen Syntax oder Semantik dieser Oktette keine Rolle: Bits sind Bits.“ ("End-to-end services are concerned with data transfer. Unlike the application services, which are concerned with information transfer, end-to-end services are interested solely in moving octets (eight-bit values) from one system to another. The syntax and semantics of those octets are unimportant: bits are bits.") [The Open Book, Marshall T. Rose, Prentice Hall, Englewood Cliffs, N.J. 1990, S. 35.]

Anwendungsebene	Anwendungsprotokoll	Anwendungsebene
Präsentationsebene	Darstellungsprotokoll	Präsentationsebene
Sitzungsebene	Kommunikations-steuerungsprotokoll	Sitzungsebene
Transportebene	Transportprotokoll	Transportebene
Netzwerkebene	Vermittlungsprotokoll	Netzwerkebene
Verbindungsebene		Verbindungsebene
Physikalische Ebene		Physikalische Ebene

Datentransfer

Abbildung 3-2: Schema der Kommunikation unter Verwendung des OSI-Schichtenmodells. Die eigentlichen Daten werden von der jeweiligen physikalischen Schicht an die nächste weitergereicht. Der Weg der Verbindung beginnt beim Rechner auf der linken Seite in der Anwendungsebene. Die Daten werden nach unten weitergereicht, sie durchlaufen dabei verschiedene Software-Ebenen des sendenden Rechners und durchlaufen dann analog diese Ebenen in der umgekehrten Reihenfolge auf dem empfangenden Rechner. Die einzelnen Protokolle sind dabei so aufeinander abgestimmt, daß die Daten in der jeweiligen Schicht richtig aufbereitet und interpretiert werden.

OSI-Dienste

FTAM (File Transfer Access and Management)

Dem Wunsch der Anwender nach problemlosem Dateiaustausch steht die Produktvielfalt auf dem Rechnermarkt mit unterschiedlichen Betriebssystemen und heterogenen Netzwerken gegenüber. Die diversen Betriebssysteme, wie z.B. BS2000, MVS, VMS, UNIX (SINIX) und MS-DOS unterscheiden sich in der Methode der Dateiverwaltung, in der Art des Dateizugriffs, durch die Satzstruktur usw. so, daß diese Strukturen nicht von einem Rechner auf den anderen abgebildet werden können. Zu diesen betriebssystembedingten Besonderheiten für eine offene Kommunikation kommen die herstellerspezifischen Kommunikationsarchitekturen, wie z.B. NEA, SNA und DNA. In der FTAM-Norm (ISO-Norm 8571 aus dem Jahr 1988) wird ein international gültiger Standard für Dateiübertragung, Dateizugriff und Dateiverwaltung festgelegt. Damit werden die folgenden dateibezogenen Aktivitäten abgedeckt:

- Dateiübertragung über netzweit gültige virtuelle Dateien
- Dateizugriff für Workstations ohne eigene Festplatte
- gemeinsame Ausgabe auf Drucker, Spoolsystem usw.
- Zugriff auf ferne Datenbanken

Eine virtuelle Datei ist eine Sammlung von Dateien, vergleichbar mit einem Dateisystem eines Betriebssystems. Um mit FTAM arbeiten zu können, muß ein System eine besondere Schnittstelle anbieten. Um mit FTAM arbeiten zu können, benötigt ein Betriebssystem eine Schnittstelle zwischen den Dateisystem-Diensten und den FTAM-Diensten. FTAM überprüft eine Datei auf zwei Komponenten. Dies sind zum einen Attribute, die Informationen über die Datei geben, und zum anderen der Dateiinhalt selbst. Eine Datei ist danach inhaltlich einer der fünf folgenden Gruppen zuzuordnen:

FTAM-1	Unstrukturierte Textdatei
FTAM-2	Sequentielle Textdatei
FTAM-3	Unstrukturierte Binärdatei
FTAM-4	Sequentielle Binärdatei
FTAM-5	Einfache hierarchische Datei

X.400

Der OSI-Standard für E-Mail (Verwaltung von elektronischer Post) ist X.400. Man hat von X.400 sogar als von einer „OSI-Flaggschiff-Anwendung“ gesprochen [Rose, S. 463]. Im Jahr 1984 hat CCITT eine Reihe von Empfehlungen zur Verwaltung von Meldungen veröffentlicht. Im Jahr 1988 entwickelten hierzu CCITT und ISO gemeinsam eine Empfehlung. X.400 besteht aus mehreren Schichten. Die äußerste Schicht ist eine Schnittstelle (User Agent) zwischen dem Menschen als Benutzer und dem E-Mail-System. Diese kommuniziert mit einer Vermittlungsstelle (Message Transfer System), die aus einem oder mehreren Message Transfer Agents besteht. Ein weiteres Element bildet der Message Store, eine Art vermittelnder Zwischenspeicher zwischen dem User Agent und dessen lokalem Message Transfer Agent. Ein User Agent holt die elektronische Post eines Benutzers aus dem Zwischenspeicher (Message Store).

Eine Nachricht im X.400-Format besteht aus zwei Teilen: einer Art Umschlag und der Nachricht, die in dem Umschlag steckt. Der Umschlag enthält Informationen über den Absender, den Verteiler und einige zusätzliche Informationen, die für die Zustellung nötig sind. Die eigentliche Nachricht enthält normalerweise einen Kopf mit Absender, Empfänger, Betreff, Sendezeit und einen oder mehrere Textteile. Da der sogenannte Textteil nicht unbedingt Text enthalten muß, ist das X.400-Protokoll sehr flexibel und zukunftsorientiert, denn es können auch Faxe, Tonaufzeichnungen oder Bilder enthalten sein. Das Entwicklungsprojekt Mercury von Siemens Nixdorf ist ein E-Mail-System mit Multimedia-Fähigkeiten, das sowohl mit internet als auch X.400 betrieben wird (siehe Abschnitt „Notwendige Infrastruktur” auf Seite 232).

X.500

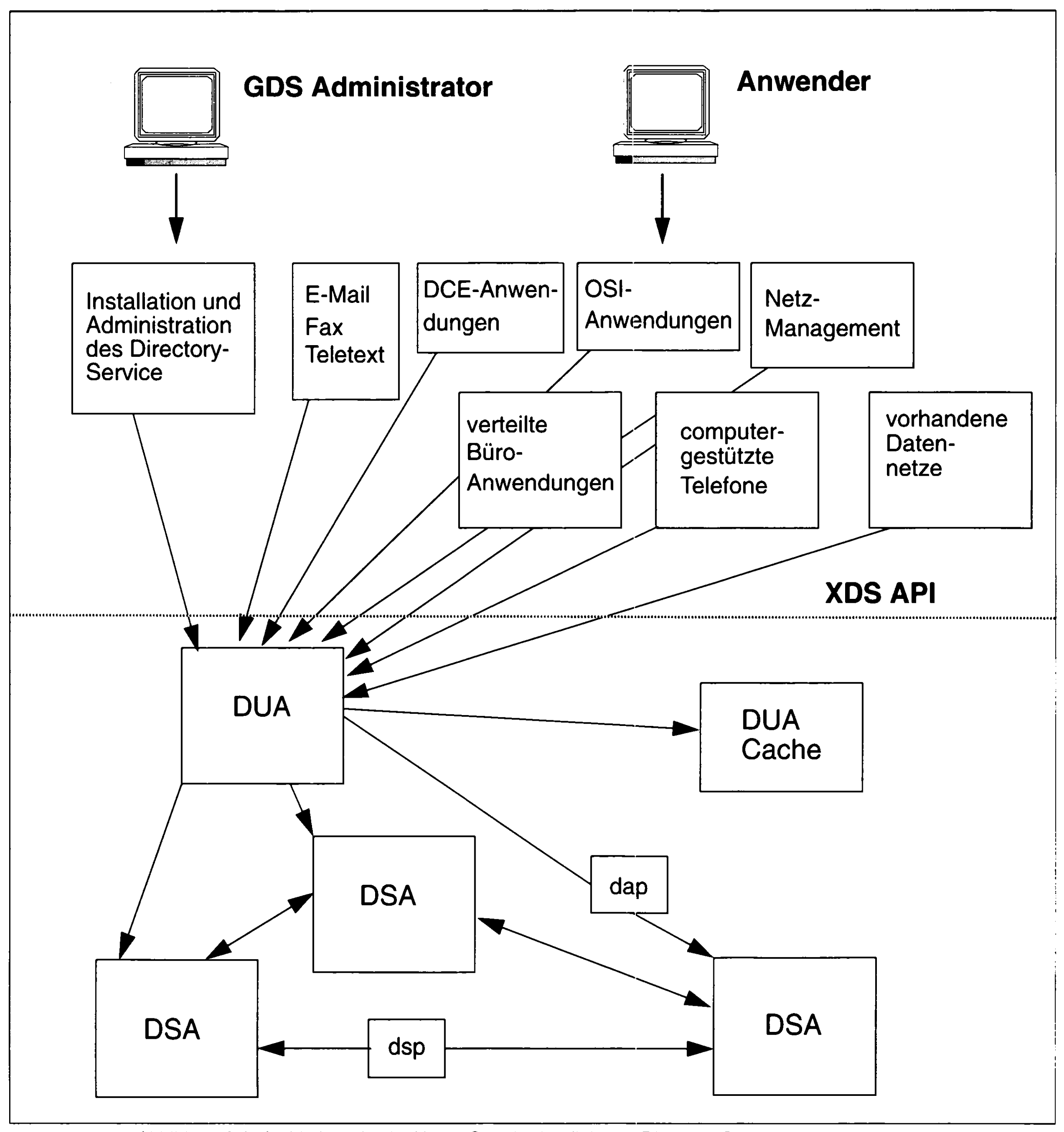


Abbildung 3-3: Architektur des im X.500-Standard definierten Directory-Systems.

Das elektronische Teilnehmerverzeichnis nach dem X.500-Standard bietet die Möglichkeit, Namen von Objekten auf computer- bzw. netzwerkgerechten Adressen abzubilden und die Eigenschaften solcher Objekte zu speichern. Diese Objekte können Computer, Eingabe-/Ausgabegeräte, Dateien, Anwendungen und Netzelemente und auch natürliche oder juristische Personen sein. Dieses Teilnehmerverzeichnis läßt sich am besten mit den sogenannten weißen und gelben Seiten des Telefonbuchs vergleichen. Solche Verzeichnisse haben allerdings den Nachteil, daß sie unterschiedliche Besitzer und dafür Verantwortliche haben und häufig redundante oder inkonsistente Informationen beinhalten. Demgegenüber stellt das Teilnehmerverzeichnis nach X.500 ein intelligentes, einheitliches und standardisiertes Kommunikations- und Eigenschaftsverzeichnis dar.

Abbildung 3-3 zeigt die im X.500-Standard definierte Architektur des Directory-Systems. Es besteht aus Zugangsbausteinen (Clients), genannt **D**irectory **U**ser **A**gents (DUA), und den Erbringern von Leistungen (Server), genannt **D**irectory **S**ystem **A**gents (DSA). Die Directory User Agents befinden sich auf Client-Rechnern, die Directory-Services befinden sich auf den Server-Rechnern. DUA und DSA kommunizieren über das Directory Access Protocol (DAP), die DSAs kommunizieren untereinander über das Directory System Protocol (DSP). Nutzer des Directory Service können E-Mail, Dateitransfer oder verteilte Büroanwendungen usw. sein. Zu diesem Zweck bietet der Directory User Agent, ein Anwendungsprozeß im Client-Rechner, eine Anwenderschnittstelle. Diese Anwenderschnittstelle (XDS: API Directory-Service), von X/Open definiert, steht auch dem Management des Directory-Service für Installations- und Verwaltungszwecke zur Verfügung. Die Trennung von Client und Server gestattet es, neue Clients problemlos hinzuzufügen. Darüberhinaus ist es ohne Rückwirkung auf das Gesamtsystem möglich, den Server-Rechner durch einen anderen zu ersetzen oder seinen Aufstellungsort zu verändern.

Integration von PCs in Netzwerken

Ursprünglich waren PCs nur dazu gedacht, die für einen Arbeitsplatz typischen PC-Anwendungen, Laufwerke und Drucker bereitzustellen. Aufgrund der Möglichkeiten, die PCs heute bieten (Rechnerleistung, Speicherkapazität, grafische Bedienoberfläche, individuelle Einstellung der persönlichen Arbeitsumgebung, Verfügbar-

keit usw.) sowie ihrer steigenden Verbreitung, ist es durchaus sinnvoll, PCs in bestehende IT-Lösungen der Unternehmen zu integrieren, bzw. neue IT-Konzepte unter Einbeziehung von PCs zu realisieren.

Die Integration von PCs ist in loser, aber auch in sehr enger Form möglich. Grundsätzlich kann man drei Stufen der Integration unterscheiden, wobei die Grenzen zwischen diesen Stufen fließend sind.

- PCs können zur Terminalemulation verwendet werden, um den Zugang zu anderen Computer-Ressourcen zu ermöglichen, wie z.B. ein Mehrplatz-System oder einen Großrechner. Dies kann die Emulation eines 97801-Terminals, eines X-Terminals oder eines 3270-Terminals sein. Terminalemulationen bieten die Möglichkeit, schon vorhandene Terminals durch PCs zu ersetzen und auf diesen PCs parallel dazu PC-Standardsoftware und schon existierende Anwendungen einzusetzen. Über entsprechende Multiplex-Mechanismen können auch mehrere Sitzungen parallel durchgeführt werden. Der Einsatz von Terminalemulationen ist zudem der erste Schritt, einen allmählichen Übergang von „konventionellen“ Anwendungen auf Client-Server-Anwendungen zu erreichen.
- PCs können an ein LAN angeschlossen werden und sich auf diese Weise Ressourcen wie Drucker oder zentral auf Platte gehaltene Daten teilen. Wenn ein Midrange-Rechner in das LAN eingebunden wird, kann dieser als Server für unterschiedliche Ressourcen eingesetzt werden, wie z.B. Dateien, Drucker, Datenbanken oder Netzverbindungen. Diese Ressourcen werden durch öffentlich zugängliche Dienste (z.B. File Transfer, SNA Gateways), verteilte Dateisysteme (z.B. NFS, DFS) oder Netzwerk-Betriebssysteme (wie z.B. LM/X) zur Verfügung gestellt. Diese Strategie bietet unter anderem die Möglichkeit, Daten zentral zu verwalten (Server oder Host) und für PC-Anwendungen zugänglich zu machen. Die Daten können lokal weiter bearbeitet und die Arbeitsergebnisse auf einem zentral angeschlossenen Drucker ausgegeben werden.
- Darüber hinaus stoßen wir in Bereiche vor, die den wirklichen verteilten Systemen zuzurechnen sind.

Die Einbindung von PCs in Netze wird in weiteren Kapiteln dieses Buchs dargestellt (siehe Abschnitt „Integration von SINIX- und PC-Netzen mit LM/X” auf Seite 128 und Abschnitt „PC-Integration” auf Seite 175).

Netzwerkprodukte von Siemens Nixdorf

TRANSIT

Siemens Nixdorf wird für den Kundenkreis mit SNA-Netzen weiterhin die Produkte der TRANSIT-Familie anbieten. Dazu gehören Produkte, die SINIX-Systeme in die SNA-Umgebung einbinden. Siemens Nixdorf-Produkte sind in der Lage, SNA-Netzsegmente vollständig zu ersetzen. Dabei werden SINIX-Rechner als SNA-Gateways eingesetzt und erhöhen so die Leistungsfähigkeit beträchtlich. Die Produkte der TRANSIT-Familie unterstützen 3270-Emulationen, SNA-Programmschnittstellen, wie z.B. CPI-C, EHLLAPI, LU0API, Eingangs- und Service-Schnittstellen für NetView, SNA Server, SNA Passthrough, Zugang zu SNA-Netzen mittels SDLC, X.25 und Token Ring.

SINIX-Systeme können auf verschiedene Weise in SNA-Netze eingebunden werden:

- SINIX-Systeme können mit einem oder mehreren IBM-Systemen zu einem Netzverbund zusammengeschlossen werden.
- SINIX- und MS-DOS-Netzwerksegmente können mit Hilfe eines SINIX-Servers in ein SNA-Netz eingebunden werden. Diese Netzwerksegmente können hierarchisch verbunden werden. Auf diese Weise wird es möglich, eine Reihe von Netzwerksegmenten mit einem SNA-Netz zu verbinden.
- SINIX-Systeme können mit anderen Systemen oder Netzwerksegmenten auf derselben hierarchischen Ebene zu einem Peer-to-peer Netzverbund zusammengeschlossen werden.

NEA

Das Produkt von Siemens Nixdorf, Network Environment Architecture (NEA), wird selbstverständlich von SINIX-Systemen unterstützt. Die Produkte der TRANSDATA-Familie erhielten über die Jahre einen größeren Funktionsumfang und unterstützen sowohl NEA als auch die Internet-Protokoll-Familie und die OSI-Protokoll-Familie.

Die ersten TRANSDATA-Netzwerke wurden schon im Jahr 1974 eingerichtet. Sie waren ursprünglich für homogene Netze gedacht, wurden aber schrittweise erweitert. Damit konnten auch Systeme anderer Hersteller eingebunden und zusätzliche Protokoll-Architekturen unterstützt werden. So ist es z.B. möglich, Netzwerke in TRANSDATA-Norm mit SNA-Netzwerken zusammenzuschließen.

FT/FTOS

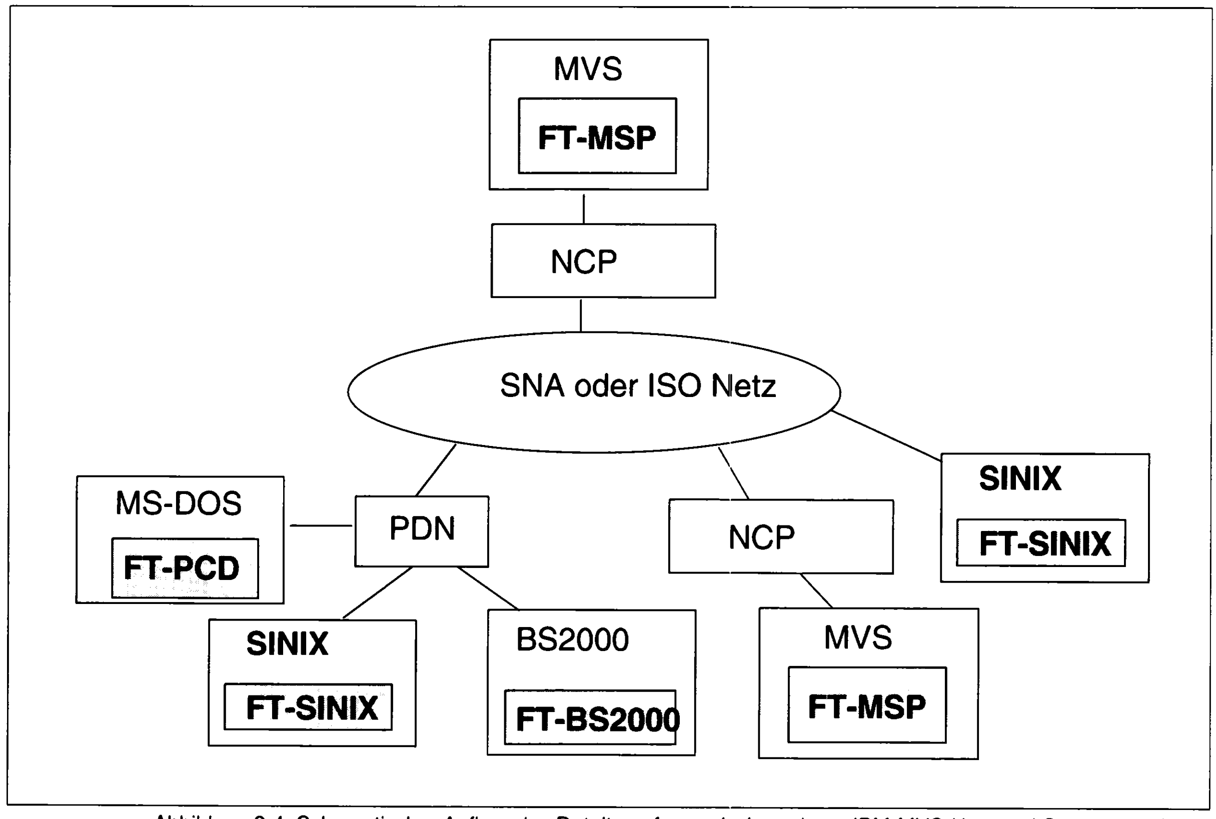


Abbildung 3-4: Schematischer Aufbau des Dateitransfers zwischen einem IBM-MVS-Host und Systemen mit SINIX, MS-DOS und BS2000 bei Verwendung der Produkte der FT-Familie von Siemens Nixdorf.

Siemens Nixdorf bietet schon seit einigen Jahren File-Transfer-Produkte (FT) an. Diese ermöglichen den Datenaustausch mit einer Vielzahl von unterschiedlichen Rechnern bei hoher Leistung und breitem Funktionsumfang. Seit FTAM, die OSI-Norm für den File Transfer, verfügbar ist, war es die Strategie von Siemens Nixdorf,

einerseits den OSI-Standard zu erreichen, andererseits aber den größeren Funktionsumfang unserer schon vorhandenen File-Transfer-Produkte zur Verfügung zu stellen. Die Produkte gemäß Siemens-Nixdorf-Normen sind tausendfach installiert und bieten für die nächste Zukunft einen höheren Leistungsumfang als die entsprechenden Produkte gemäß FTAM-Norm. Dazu gehören z.B. Folgeverarbeitung im entfernten System, automatischer Wiederanlauf und Datenkomprimierung.

Der File-Transfer über das FTAM Protokoll (ISO 8571) wird für SINIX-Systeme mit FTOS-SINIX durchgeführt, während der File-Transfer über NEA von Siemens Nixdorf mit FT-SINIX durchgeführt wird. Dabei wurde FTOS so in die bestehende Produktumgebung eingebunden, daß sowohl Koexistenz als auch Migration möglich sind. Dieser Ansatz bringt für neu gewonnene Anwender und für schon bestehende Systeminstallationen viele Vorteile.

Siemens Nixdorf bietet für die eigenen Betriebssysteme (z.B. unter BS2000, SINIX, TOS, AMBOSS und MS-DOS/WINDOWS) und für IBM-Rechner (unter MVS) FT-Produkte an. Da die Siemens-Nixdorf-FT-Normen offengelegt und frei zugänglich sind, hat die Digital Equipment Corporation (DEC) für ihre VAX-Systeme unter VMS und ULTRIX mit FTSIE eigene FT-Produkte entwickelt. Dadurch können mit Systemen von Siemens Nixdorf Daten leicht ausgetauscht werden. Neben diesen bieten Softwarehäuser weitere FT-Produkte an, speziell für die Betriebssysteme OS/2 und MS-DOS. Auf eine einheitliche Funktionalität sowie eine einheitliche Benutzerschnittstelle innerhalb der FT-Produktfamilie wurde großer Wert gelegt. FT-SINIX bietet unter anderem:

- Symmetrische Übertragungsinitiative
- Auftragsspeicherung
- Automatischen Wiederanlauf
- Datenübertragung mit Folgeverarbeitung
- Datenkomprimierung
- File-Management-Funktionen zur Dateiverwaltung
- Ergebnismitteilung
- Sicherheitsfunktionen
- Kommunikation mit IBM-VMS-Hosts
- Übereinstimmung mit der X/Open BSFT-Komponente

Um in einem großen und weiträumigen Rechnerverbund mit teilweise fehleranfälligen Übertragungswegen einen File-Transfer sinnvoll betreiben zu können, ist die Verfügbarkeit von größter Bedeutung. Die Produkte der FT-Familie bieten diese Eigenschaften durch komfortable Auftragsspeicherung und automatischen Wiederanlauf.

DIR-X

DIR-X ist das elektronische Teilnehmerverzeichnis für SINIX-Systeme nach dem X.500 Standard von Siemens Nixdorf. DIR-X arbeitet auf INFORMIX- und C-ISAM-Datenbanken. Eine MOTIF-Version ist in Kürze verfügbar. Produktversionen für die Betriebssysteme BS2000 und MS-DOS sind in Entwicklung.

Stand-alone-Anwendungen und verteilte Anwendungen aller Art können über die von X/Open genormte Schnittstelle XDS auf das Directory-System zugreifen.

Bei der Konzeption und Entwicklung von DIR-X wurden die folgenden Ziele gesteckt und erreicht:

- netzweite Verfügbarkeit und Eindeutigkeit der Daten (die Datenkonsistenz wird z.B. dadurch gewährleistet, daß Informationen jeweils nur an einer Stelle geändert werden können)
- flexible Datenstrukturen
- Unterstützung benutzerfreundlicher Namensgebung
- Einsetzbarkeit für verschiedene Anwendungen
- Unterstützung verschiedener Suchabfragen
- einheitliche Zugangsberechtigungen und Zugangsüberprüfungen
- Konformität zum X.500-Standard
- Konformität zum X/Open XPG4 Directory API
- einfache Verwaltung
- Portabilität bzw. Verfügbarkeit auf mehreren Plattformen
- Kompatibilität mit vorhandenen Netzwerken

Mit diesen Eigenschaften bietet der Directory Service von Siemens Nixdorf eine wichtige Voraussetzung für einen gut funktionierenden unternehmensinternen und -externen Informationsaustausch. Dabei wird die ständige Aktualität und Konsistenz von Adreßinformationen gesichert. Den Nutzern dieser Dienste bleibt dabei die Komplexität der über ein Netz verteilten Dienste verborgen. Darüber hinaus wirkt der Directory Service als Integrationsfaktor für unterschiedliche Dienste, Anwendungen, Organisationseinheiten usw. Wie hoch die Akzeptanz des Siemens-Nixdorf-Produkts DIR-X mittlerweile geworden ist, zeigt sich daran, daß DIR-X von OSF als Global Directory Service (GDS) für das Distributed Computing Environment (DCE) ausgewählt wurde. Das macht den Siemens-Nixdorf-Directory-Service nach X.500 zur Industrienorm.

CMX

CMX ist der SINIX-Kommunikations-Manager. CMX stellt SINIX-Anwendungen eine Schnittstelle zur Verfügung, die von den zugrundeliegenden Transportdiensten unabhängig ist. Alle nötigen Anpassungen an die unterschiedlichen zugrundeliegenden Transportdienste werden von CMX durchgeführt. CMX unterstützt eine Reihe von Transport-Schnittstellen (API), z.B.:

- XTI, Transport-Schnittstelle von X/Open nach XPG4
- TLI, Transport-Schnittstelle von UNIX SVR4
- sockets, Internet-Transport-Schnittstelle
- ICMX, Standard-Transport-Schnittstelle von Siemens Nixdorf

4 Benutzerumgebung

Überblick

Computeranwender haben meist wenig Interesse an den Technologien und Mechanismen, die einem Computer zugrunde liegen. Anwender möchten eine Aufgabe erledigen und erwarten vom Computer, daß er sie dabei unterstützt, produktiver zu arbeiten. Einige Kriterien für ein Computersystem sind aus der Sicht des Anwenders

- leichte Bedienbarkeit
- einheitliche Oberfläche
- überzeugende Leistung
- minimaler Lernaufwand

Anwender bevorzugen eine einheitliche Bedienoberfläche für unterschiedliche Anwendungen, die auf einem Rechner laufen. Anwender, die mit mehr als einem Rechnertyp arbeiten, wünschen gleiche Bedienoberflächen für verschiedene Hardwareplattformen. Ein Anwender, der vorwiegend auf einem SINIX-System arbeitet, gelegentlich jedoch einen MS-Windows PC benutzt, hätte gerne auf beiden Rechnern dieselben Fenstertypen, Menüs, usw.

Da Bedienoberflächen für Anwendungsprogramme nicht nur von Siemens Nixdorf realisiert werden, sondern auch von Softwarehäusern und hausinternen Entwicklungsabteilungen beim Kunden, beschreibt dieses Kapitel die Werkzeuge, die SINIX für die Entwicklung von Bedienoberflächen zur Verfügung stellt, sowie die Charakteristika der Bedienoberflächen, die mit diesen Werkzeugen erstellt werden können. Es wurde viel Mühe auf Werkzeuge und Anwendungshilfen verwandt, die Siemens Nixdorf anbietet, um gute, benutzerfreundliche Bedienoberflächen für Anwendungsprogramme bereitzustellen, die unter SINIX entwickelt wurden und unter SINIX ablauffähig sind.

Im vorliegenden Kapitel werden zuerst grafische Bedienoberflächen vorgestellt, im Anschluß daran die eher traditionellen alphanumerischen Oberflächen — solche, die für Eingaben und Anzeigen auf die Tastaturzeichen (Buchstaben und Zahlen) begrenzt sind. Zudem wird ein im Entstehen begriffener Standard zu diskutieren sein, der konzipiert wurde, um auch für alphanumerische Bildschirmarbeitsplätze ein Fenstersystem bereitzustellen. Das Kapitel schließt mit einer Besprechung der Werkzeuge zur Internationalisierung von Bedienoberflächen auf SINIX-Systemen.

Grafische Bediensysteme

In Vorwegnahme des Trends zu grafischen Bediensystemen wurde SINIX zu einem frühen Zeitpunkt seiner Entwicklung mit der COLLAGE-Fenstertechnik für Ein- und Mehrplatzsysteme ausgestattet. Um der Standardisierung gerecht zu werden, die seitdem im Bereich der grafischen Bediensysteme eingesetzt hat, unterstützt SINIX jetzt X Window und OSF/Motif. Grafische Bediensysteme werden unter SINIX zudem durch das Produkt SINIX/windows unterstützt, das auf X Window und OSF/Motif aufbaut.

Nach Untersuchungen der Marktforschungsinstitute XBusiness und Dataquest zeichnet sich der Trend zu den GUIs (**G**raphical **U**ser **I**nterfaces), den grafischen Bediensystemen, klar ab. OSF/Motif hat sich danach als „das" grafische Bediensystem unter UNIX durchgesetzt. Aus diesem Grund wurde OSF/Motif zur strategischen Schnittstelle für grafische Bediensysteme und auch im SINIX API festgeschrieben. Erst kürzlich wurde OSF/Motif von COSE (**C**ommon **O**pen **S**oftware **E**nvironment), einer Gruppe, in der die wichtigsten UNIX-Anbieter wie HP, IBM, SunSoft u.a. vertreten sind, als gemeinsam unterstützte Basis für grafische Bedienoberflächen festgelegt.

OSF/Motif steht seit 1990 auf SINIX-Systemen zur Verfügung. Da Motif im OSF-Original im wesentlichen nur aus einem Fensterverwalter und Elementen (Widgets) für die Gestaltung von Oberflächen besteht, erfolgte schrittweise eine Erweiterung um notwendige und für eine komplette Desktop-Umgebung noch fehlende Komponenten. X Window, OSF/Motif und Erweiterungen von Siemens Nixdorf wurden zum Produkt SINIX/windows zusammengefaßt, so daß die Einheitlichkeit der grafischen Bediensysteme auf allen SINIX-Hardwareplattformen sichergestellt ist.

Das X Window System

1984 standen die Computerlabors des MIT (Massachusetts Institute of Technology) vor folgendem Problem: Es wurde eine große Anzahl von Computern verschiedener Hersteller in unterschiedlichen Leistungsklassen und mit unterschiedlichen Betriebssystemen eingesetzt. Diese unterschiedlichen Computersysteme sollten in ein Netzwerk eingebunden werden, so daß alle Rechner untereinander Daten und Nachrichten austauschen konnten. Dies führte zur Idee, ein Programm, das auf einem Computer abläuft, von einem anderen Computer aus zu kontrollieren. Wenn Computer in der Lage waren, Daten auszutauschen, sollten sie auch Rechenleistung austauschen können. Schließlich leiteten diese Überlegungen die Entwicklung des X Window Systems ein.

Leistungsstarke Rechner sollten Programme ausführen, für die andere Rechner zu langsam waren. Trotzdem sollten die Benutzer dieser langsameren Rechner nicht gezwungen sein, an einen anderen Bildschirmarbeitsplatz zu wechseln. Mit diesen Überlegungen als Leitlinie entwickelte die MIT-Gruppe ein System, das die Anzeige der Ausgaben eines Computers auf einem anderen Computer ermöglichte. Programme wurden zwischen dem Frontend (dem Bildschirm und der Tastatur) und dem Backend (dem eigentlichen Rechner) aufgeteilt. Da so die Programme vieler Computer von einem einzigen Computer kontrolliert wurden, dachte man an eine Ausgabe in verschiedenen Fenstern. Um diese Funktionen bereitstellen zu können, mußte die Kommunikation zwischen zwei Computern durch ein festgelegtes Protokoll unterstützt werden. Zwischen Anwendungsprogramme und den Bildschirm wurde eine Schnittstelle gesetzt, um Anwendungen gegen unterschiedliche Grafikhardware und die Unterscheidung zwischen Frontend und Backend abzuschirmen. Bei einem Wechsel der Grafikhardware sollte nur noch die Schnittstellensoftware angepaßt werden. Da diese Schnittstelle für alle Anwendungen gleich ist, muß der anfallende Anpassungsaufwand nur einmal für jede neue Grafikhardware-Komponente geleistet werden.

Die Grundlage der Arbeit des MIT bildete das Fenstersystem „W“, das von der Stanford Universität für Testzwecke entwickelt worden war. Um das MIT-Projekt vom Stanford-Projekt unterscheiden zu können, wählte man als Namen für dieses Fenstersystem den nächsten Buchstaben im Alphabet, das „X“. Da der Schwerpunkt der Arbeit darauf lag, ein leicht portierbares System zu entwickeln, wurde einem modularen Ansatz der Vorzug gegeben. Das Projekt baute auf einem Satz von Basisfunktionen auf; neue Funktionen wurden nur in das System aufgenommen, wenn sie mit den bereits bestehenden Funktionen nicht dargestellt werden konnten.

Schon bald wurde die Entscheidung getroffen, X Window an alle Interessenten zum Preis der Kopierkosten weiterzugeben. Die Entwickler sicherten so die schnellstmögliche weite Verbreitung von X Window. Die Vorteile einer weiten Verbreitung wurden bald offensichtlich. X Window wurde schnell übernommen und andere Entwickler außerhalb des MIT leisteten wertvolle Beiträge zu seiner weiteren Entwicklung. 1986 kündigte DEC das erste kommerzielle X Window System für seine VAX Station II/GPX an.

X Window ist nicht auf UNIX-Systeme beschränkt. Es gibt X Window Server und X Clients für viele Rechner. Mittlerweile wurde die Verantwortung für die Weiterentwicklung von X Window vom MIT auf das X Konsortium übertragen. In diesem Konsortium haben sich mehr als 30 Hersteller und Organisationen zusammengeschlossen, darunter auch Siemens Nixdorf, um die weitere Entwicklung von X Window zu fördern und die Einhaltung des Standards zu überwachen.

Im Gegensatz zu MS-Windows, Presentation Manager von OS/2 und MacOS gibt es unter X Window keine Vorschriften, wie ein Programm bestimmte Aufgaben zu erfüllen hat. Diese Entscheidungen wurden bewußt den Entwicklern von X Clients überlassen, um deren Kreativität nicht einzuschränken. So kann der Entwickler eines X Client die Gestaltung der Benutzerumgebung einschließlich der Anordnung von Buttons und Menüs selbst bestimmen. Der Entwickler wird nicht durch restriktive Design-Regeln eingeengt. Andererseits könnte der Anwender durch eine Vielfalt von Designvorgaben für Bediensysteme verwirrt werden und der problemlose Umgang mit Anwendungsprogrammen behindert werden.

Abgesehen von Überlegungen, die das Copyright der Merkmale von Bedienoberflächen, ihr „Look und Feel" betreffen, kann ein individuelles Design angewandt oder aber ein bereits bestehendes Design übernommen werden. Entwicklern wird nicht vorgeschrieben, wie eine bestimmte Funktion zu implementieren ist. Es bleibt ihnen überlassen, ihren persönlichen Stil zu wählen.

OSF/Motif

Obwohl X Window als System zur Erstellung grafischer Bediensysteme im Vergleich zu anderen Systemen viele Vorteile bietet (Netzwerkunterstützung, frei konfigurierbar, hardware-unabhängig), ergaben sich sowohl für den Anwender als auch für den Entwickler ernstzunehmende Einschränkungen. Die Funktionen, die dem Entwickler durch das ursprüngliche X Window System zur Verfügung gestellt wurden, waren nicht leistungsfähig genug, und die Toolkits, die darauf aufsetzten, nur schwer portierbar. Von System zu System mußte sich der Anwender mit neuen Fensterverwaltern vertraut machen, da das Standard-Verwaltungsprogramm des MIT auf fast allen X-Systemen durch Eigenentwicklungen des jeweiligen Herstellers ersetzt wurde. Dies führte für den Anwender zu einer eher verwirrenden Auswahl an verschiedenen Bediensystemen, die alle unterschiedlich aussahen, im Grunde jedoch identisch waren.

Im Juli 1988 führte OSF eine Technologieausschreibung, ein Request for Technology (RFT), für grafische Bediensysteme durch. Grundvoraussetzung dieses RFT war, daß das System auf dem X Window System aufbauen sollte. Im Januar 1989 gab OSF die Ergebnisse des RFT bekannt. Ausgewählt wurden das Toolkit und die User Interface Language (UIL) von DEC, die Toolkiteinbindung, der Fensterverwalter und der 3-D Look von HP sowie das Look and Feel des Presentation Manager von Microsoft. Das Ergebnis erhielt den Namen Motif.

Mit OSF/Motif erhält der Entwickler eine gemeinsame, portable Programmierschnittstelle (API - Application **P**rogramming **I**nterface) auf einer Vielfalt von Hardwareplattformen (VAX, SPARC, MIPS, Intel, Motorola) und im Grunde genommen auf allen Systemen, die X Window unterstützen.

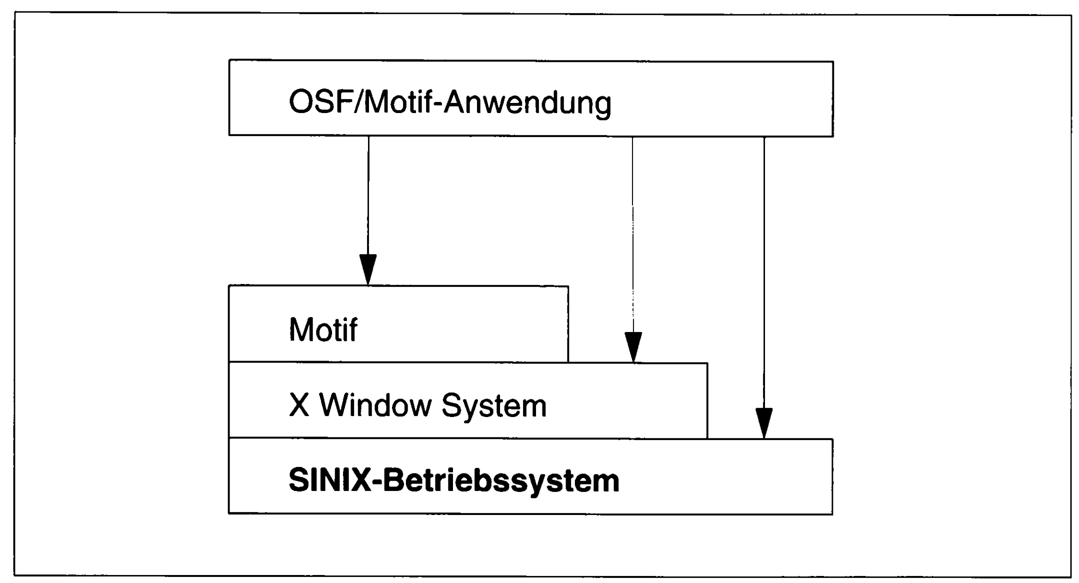


Abbildung 4-1: OSF/Motif-Anwendungen benutzen die Dienste, die OSF/Motif, X Window und das SINIX-Betriebssystem anbieten.

Das Erscheinungsbild einer Anwendung, die auf OSF/Motif basiert, ähnelt einer auf MS-Windows basierenden Anwendung. Dies ermöglicht dem Anwender einen gleitenden Übergang von PC-basierten Systemen auf SINIX-Systeme. Auch wenn der Entwickler seinen MS-Windows- oder OS/2-Code nicht direkt übernehmen kann, kann zumindest das Erscheinungsbild des Programms mit minimalen Änderungen übernommen werden. Da auf dem X Window System aufgesetzt worden war, wurde für OSF/Motif- und OSF/Motif-basierte Anwendungen leichte Portierbarkeit erreicht. Da X Window sehr portabel und auf einer breiten Palette von Systemen verfügbar ist, konnte sich OSF/Motif schnell weit verbreiten. Neben seiner Verfügbarkeit unter UNIX wird OSF/Motif von etwa 38 Betriebssystemen unterstützt, darunter OS/2, MS-Windows und Macintosh. Natürlich arbeiten OSF/Motif-Anwendungen nicht nur mit OSF/Motif zusammen, sondern auch mit dem Betriebssystem, so daß hier der problemlosen Portierbarkeit von OSF/Motif-Anwendungen Grenzen gesetzt sind.

OSF/Motif Elemente

OSF/Motif besteht aus drei Hauptelementen, dem Toolkit, der User Interface Language (UIL) und dem Fensterverwalter.

Toolkit

OSF/Motif bietet dem Entwickler ein standardisiertes Toolkit mit 30 Widgets und sechs Gadgets. Da die OSF/Motif Widgets auf Basis der Xt Intrinsics (MIT) erstellt wurden, können Software-Entwickler dem System spezielle Widgets für besondere Aufgaben nahtlos hinzufügen.

UIL

Mit der User Interface Language kann das Bediensystem eines Programms beschrieben werden, ohne daß dessen eigentliche Funktionen bekannt sind. Zusätzlich wird die UIL dazu verwendet, festzulegen, welche Funktionen eines Programms als Reaktionen auf bestimmte Aktionen des Anwenders ausgeführt werden. Aus dieser Beschreibung erstellt der UIL-Compiler eine Ressourcendatei, die vom Programm geladen wird.

Fensterverwalter

Der OSF/Motif Fensterverwalter legt einen Rahmen um die einzelnen Fenster, mit dem die Fenster jeweils verschoben oder in ihrer Größe verändert werden können. Der Fensterverwalter richtet sich dabei nach der Funktionalität von MS-Windows oder Presentation Manager (OS/2). Zu einem Sinnbild (Icon) verkleinerte Fenster werden in einem speziellen Iconfenster angezeigt.

SINIX/windows

SINIX/windowsgliedert sich in zwei Bestandteile: das SINIX/windows User Environment (SUE) und SINIX/windows Development (SD). SD ist mit der sogenannten OSF/Motif Validation Test Suite getestet worden und hat das Gütesiegel „OSF/ Motif Validated" erhalten.

SINIX/windows User Environment

SUE bietet dem Anwender einen komfortablen elektronischen Schreibtisch zur Bedienung des Betriebssystems und eigener Anwendungen. Zusätzlich wird dem Anwender umfangreiches Zubehör angeboten, das ihn bei der Einrichtung einer effektiveren Arbeitsumgebung unterstützt. Im einzelnen besteht das SUE aus folgenden Komponenten: dem OSF/Motif Runtime-System, einem elektronischen Schreibtisch, dem Desktop, Konfigurationshilfen zur benutzerspezifischen Anpassung und einem Hypertext-ähnlichen Online-Hilfesystem. OSF/Motif bietet ein Bediensystem, dessen Erscheinungsbild und Bedienbarkeit MS-Windows ähnelt. Dadurch verfügen Anwender über ähnliche Bediensysteme auf SINIX-Systemen und MS-Windows-Systemen. Die Fenstertechnik ermöglicht das gleichzeitige Arbeiten mit mehreren Anwendungen. So wird bei der Bearbeitung komplexer Vorgänge Zeit gespart. Zusätzlich stehen Oberflächenobjekte eines Bediensystems wie Buttons, Pulldown-Menüs oder Pop-up-Menüs zur Verfügung. Die Cut-Copy-Paste-Funktionalität ermöglicht dem Anwender, beliebige Texte oder Symbole zu markieren, sie temporär zu speichern und an einer anderen Stelle wieder einzufügen. So werden dem Anwender mühselige Tastatureingaben erspart und Tippfehler vermieden.

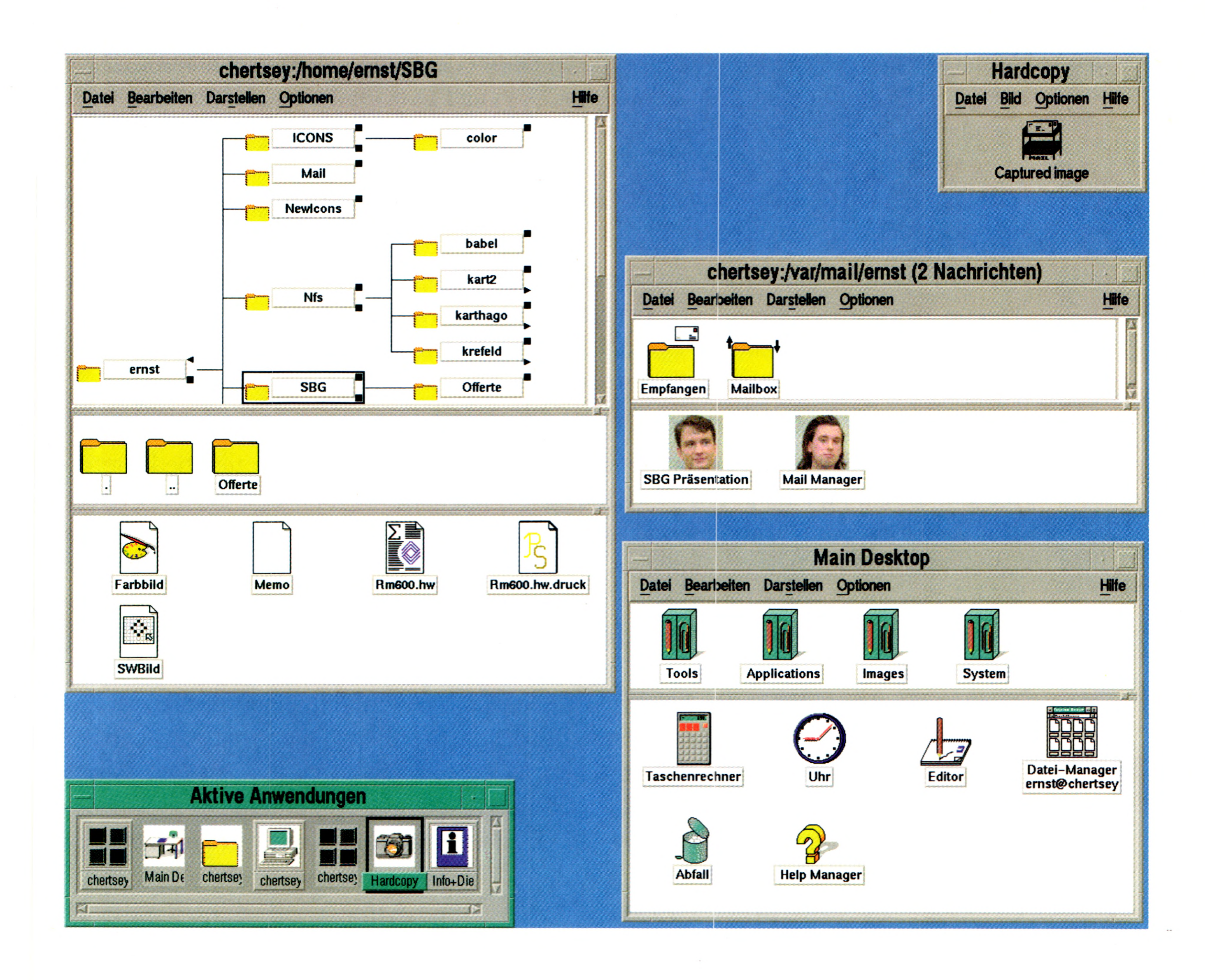


Abbildung 4-2: Moderne grafische Bediensysteme wie SINIX/windows stellen dem Anwender einen komfortablen elektronischen Schreibtisch zur Betriebssystembedienung und für seine eigenen Anwendungen zur Verfügung.

Mit dem Desktop können häufig wiederkehrende Bedienvorgänge mit der Maus grafisch interaktiv ausgeführt werden. Dies ersetzt komplexe Eingaben über die Kommandozeile der Shell. Beliebige Anwendungen können interaktiv in den Desktop eingebunden oder neue Dateitypen definiert werden. Diese werden visuell durch Symbole (für Dateien und Dateiverzeichnisse) und Icons (für Programme) repräsentiert. So können Anwendungen einfach entweder durch Doppelklick mit der

Maus auf das entsprechende grafische Symbol oder mittels der sogenannten „Drag and Drop"-Funktion aufgerufen werden. Drag and Drop bedeutet, daß das grafische Symbol für eine Datei mit der Maus zu der dazugehörigen Anwendung gezogen wird (engl. to drag) und dort abgelegt wird (engl. to drop), so daß die Anwendung mit dem aktuellen Dokument direkt gestartet oder eine bereits laufende Anwendung mit neuen Daten versorgt wird.

Der Anwender kann seinen eigenen Desktop selbst interaktiv einrichten, indem er eigene Werkzeuge oder eigene Dateitypen definiert. Darüber hinaus kann der Systemverwalter den Standard-Desktop über eine umfangreiche Konfigurationssprache an die individuellen Anforderungen des Anwenders anpassen.

Für die gesamte Bedienoberfläche des Desktop steht ein hypertext-ähnliches Hilfesystem über die Menüfunktion *Hilfe* zur Verfügung. Hypertext-ähnlich bedeutet in diesem Zusammenhang, daß Hilfefenster so angeordnet sind, daß der Anwender sich jeweils detailliertere Informationen zu einem speziellen Thema anzeigen lassen kann, indem er ein hervorgehobenes Wort anklickt. Der Anwender kann so jederzeit kontextsensitive Hilfe-Informationen erhalten. Er klickt mit der Maus auf ein bestimmtes Thema (entweder ein Symbol im Fenster oder einen Menüeintrag) und der entsprechende Hilfetext erscheint sofort. Das heißt, es muß kein Index zur Hilfe-Dokumentation benutzt werden, um die Information zum aktuellen Problem zu finden, sondern die benötigte Hilfe kann direkt im Zusammenhang mit dem aktuellen Arbeitsvorgang abgefragt werden. Unter SUE wird das Arbeiten mit OSF/Motif durch Anwenderfunktionen zur SINIX-Systembedienung erleichtert. Diese Anwenderfunktionen, die in Form leicht verständlicher Icons visualisiert werden, bestehen aus einer Reihe von Werkzeugen.

Das Symbol *Desktop-Verwalter* ermöglicht dem Anwender, Symbole für alle häufig benötigten Programme oder Werkzeuge ständig sichtbar auf dem Bildschirm zu haben. Alle anderen Dateien und Programme sind über den Dateiverwalter verfügbar, der diese grafisch repräsentiert. Alle Werkzeuge, die nicht standardmäßig auf dem Desktop liegen, aber jederzeit aufrufbar sind, befinden sich im Werkzeugkatalog. Von dort aus kann der Anwender sie bei Bedarf starten oder die Drop-and-Drag-Funktion der Maus benutzen, um sie auf seinen eigenen Desktop zu bewegen. Da der Desktop eine Anwendung ist, die auf OSF/Motif und dem X Window System basiert, kann der Anwender mit Hilfe des OSF/Motif-Ressource-Editors Voreinstellungen wie Farbe, Schriftart und Anordnung festlegen.

Mit Hilfe der Funktion *Info+Dienste* können jederzeit Informationen über die Arbeitsumgebung, wie etwa über installierte Software, Plattenbelegung, Rechnerauslastung und laufende Prozesse abgefragt werden. Auch Standard-Einstellungen wie das Paßwort, der verwendete Texteditor oder die Sprache, in der der Anwender mit dem System kommuniziert, können individuell von ihm selbst verändert werden. Neben dem Werkzeug „Manuale“, das dem Anwender den Zugriff auf die Online-Dokumentation ermöglicht, stehen z.B. noch eine analoge Uhr und ein Taschenrechner zur Verfügung. Mit dem Produkt werden verschiedene Editoren als Werkzeuge für den Anwender geliefert. Hierzu gehören verschiedene Texteditoren und der Ressource-Editor, der der Abfrage oder der komfortablen Änderung von OSF/Motif-Ressourcen dient. Mit Hilfe des Bitmap-Editors können grafische Symbole erzeugt werden, die beispielsweise benötigt werden, wenn benutzereigene Anwendungen in den Desktop integriert werden sollen. Ein Pixel-Editor unterstützt farbige Icons, da das SINIX/windows User Environment sowohl mit schwarz/weißer als auch mit farbiger Bedienoberfläche zur Verfügung steht.

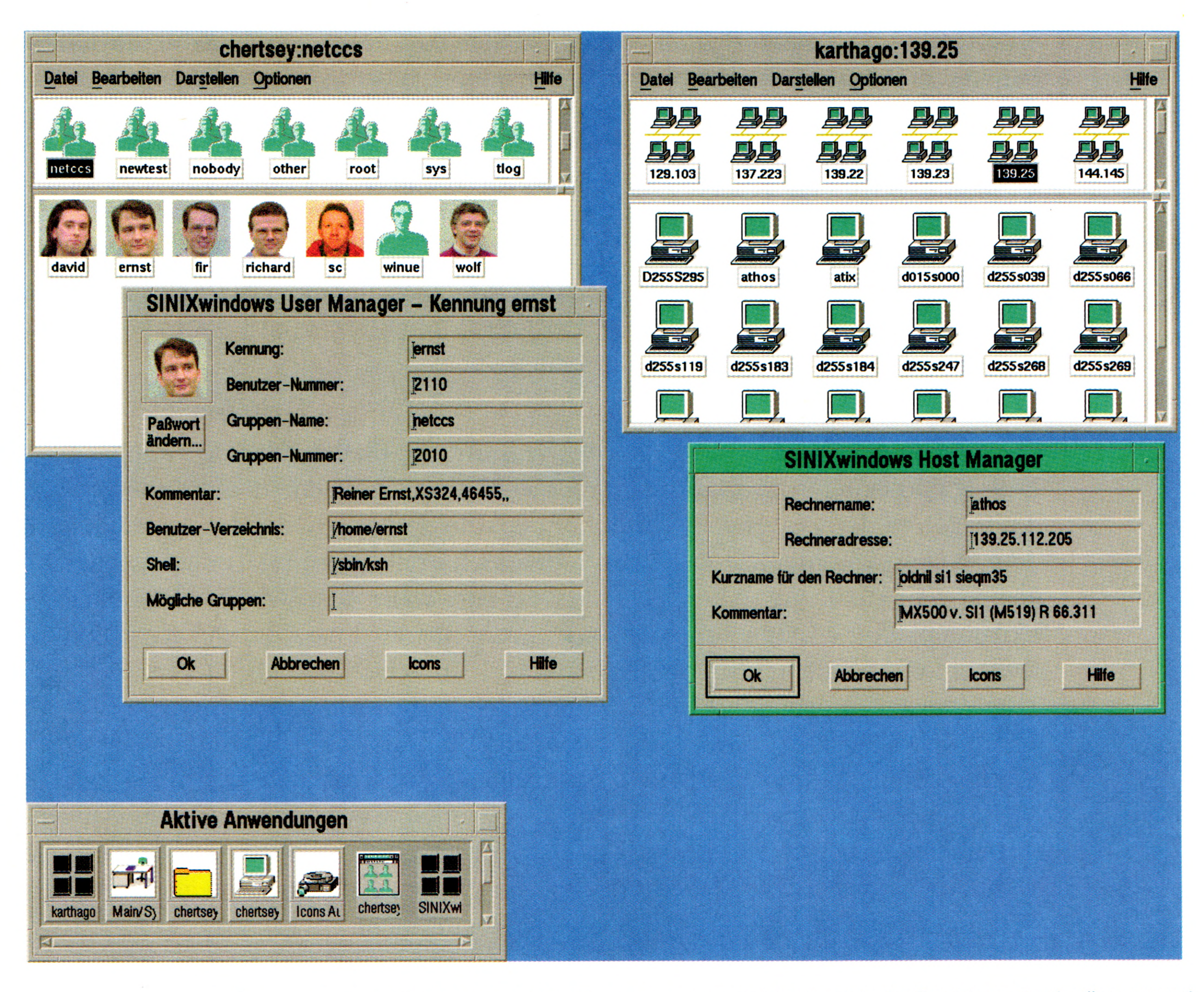


Abbildung 4-3: Das SINIX/windows User Environment ermöglicht die einfache Betriebssystembedienung und -verwaltung.

SINIX/windows Development Package

Das SINIX/windows Development Package besteht aus dem OSF/Motif-Development-System sowie Erweiterungen wie beispielsweise Widgets, die zur ursprünglichen OSF/Motif-Schnittstelle hinzugefügt wurden. Diese Erweiterungen erleichtern die Programmierung kommerzieller Anwendungen, da sich häufig wiederholende Prozeduren in Form von vier zusätzlichen Widgets für den Entwickler vorgegeben sind. Diese vier Widgets von Siemens Nixdorf werden im folgenden beschrieben:

Format-Widget

Das Format-Widget - abgeleitet vom OSF/Motif Text-Widget - erleichtert die Erstellung von Formaten unter OSF/Motif und steigert damit Produktivität und Komfort. Durch vordefinierte Feldtypen (numerisch, alphanumerisch, Datum und Zeit) wird der Programmieraufwand erheblich verkürzt, da der Entwickler die Feldtypen nicht jedesmal neu definieren muß. Bei den Feldwerten können Großbuchstaben automatisch in Kleinbuchstaben umgewandelt werden. Für alphanumerische Felder kann entweder Eingabemodus oder Überschreibmodus eingestellt werden. Eingabefehler werden durch automatische Plausibilitätsprüfungen vermieden. Format-Widgets können für Masken aller Art wie beispielsweise für Personalbögen, Auftragsbearbeitung, Kundenkarteien oder zur Terminüberwachung verwendet werden.

Tabellen-Widget

Der Entwickler kann auf einfache Weise Daten in Form von Tabellen darstellen; die Tabellen sind mit horizontalen und vertikalen Rollbalken versehen. Verschiedene Attribute können als Tabelleneigenschaften festgelegt werden: Schriftfamilien, Anzahl der Zeilen/Spalten, Feldausrichtung (linksbündig, rechtsbündig, zentriert), der Zellentyp oder die Editierbarkeit von Spalten. Zum Bearbeiten des Tabellen-Widget wird häufig das Format-Widget benutzt. Das Tabellen-Widget kann auch zur Darstellung großer Datenmengen verwendet werden (z.B. 10 000 Dateneinträge).

Hilfe-Widget

Help-Callback-Listen sind Bestandteil der Standard-Widgets von OSF/Motif. Der Hauptaufwand beim Erstellen von Hilfetexten verbleibt jedoch beim Entwickler: Festlegen des Kontexts im Programm, Auffinden des richtigen Hilfetexts im Fen-

ster, usw. Um den Aufwand zur Erstellung hypertext-ähnlicher Hilfe-Informationen drastisch zu reduzieren, bietet Siemens Nixdorf das Hilfe-Widget an. Die Unterstützung des Anwenders durch kontextsensitive Hilfe ist ein wesentliches Merkmal ergonomischer Bediensysteme. Das Hilfe-Widget stellt dem Entwickler eine vordefinierte Hilfe-Struktur zur Verfügung, die er nur noch mit den dazugehörigen Objekten auf der Bedienoberfläche verknüpfen und mit den entsprechenden Hilfetexten füllen muß.

Für den Anwender erscheint ein Hilfetext immer in einem eigenen Fenster. Falls der Bereich für die Anzeige nicht ausreichend ist, wird automatisch ein Rollbalken hinzugefügt. Im Hilfefenster stehen immer die Buttons *Ende*, *Zurück*, *Bleib* und *Hilfe* zur Verfügung. Mit Hilfe des Buttons *Zurück* springt das System in der Hierarchie der ausgewählten Querverweise um eine Stufe zurück, so daß ohne weiteren Programmieraufwand ein automatischen Blättern im Hilfetext möglich ist. Der Toggle-Button *Bleib* legt fest, ob der gerade sichtbare Hilfetext vom nächsten Hilfetext überschrieben wird oder nicht. Querverweise sind durch Fettdruck hervorgehoben und werden durch Anklicken aktiviert. Falls für ein Objekt kontextsensitive Hilfe angeboten wird, wandelt sich die Cursor-Form in ein Fragezeichen und durch Anklicken dieses Objekts in der Anwendung wird der zugehörige Hilfetext angezeigt.

Browser-Widget

Das Browser-Widget ermöglicht die Darstellung von Symbolen und Texten in einem Fenster mit der Option, einzelne oder mehrere Elemente auszuwählen und zuzuordnen. Eine mögliche Anwendung dieses Widgets ist ein elektronischer Schreibtisch (wie etwa der SINIX/windows Desktop), in dem Dateien oder Dateiverzeichnisse als Icons angezeigt werden. Das Browser-Widget bietet Funktionen zum Einfügen, Löschen und Verschieben von Einträgen.

Siemens Nixdorf Styleguide

Der Siemens Nixdorf Styleguide besteht aus einer Regelsammlung, die das „Look and Feel" von Bediensystemen beschreiben. Diese bilden die Basis für einen konsistenten und ergonomisch korrekten Ansatz für Bedienoberflächen. Eine gute Bedienoberfläche ist einfach zu handhaben. Doch gerade die Einfachheit solcher Bedienoberflächen verlangt große Sorgfalt im Erstellungsprozeß und verbirgt häufig ein komplexes Design. Eine Programmierschicht allein (z.B. Motif) läßt oft zu vieles ungenau, um eine gute Bedienoberfläche zu garantieren.

Notwendig sind präzise Richtlinien für das Layout und die Bedienabläufe der Elemente eines Bediensystems. Je detaillierter diese Richtlinien sind, umso mehr wird der Entwickler von der Last des Designs der Bedienoberfläche befreit. Der Siemens Nixdorf Styleguide enthält solche Richtlinien, die durch zahlreiche Beispiele verdeutlicht werden. Der Siemens Nixdorf Styleguide kann sowohl bei der Entwicklung neuer Bedienoberflächen als auch bei der Weiterentwicklung bestehender Bedienoberflächen angewandt werden. Er beinhaltet zudem wertvolles Übungsmaterial für den Anwender, da sich eine Vielzahl von Bediensystemen in ihrem Erscheinungsbild und ihren Bedienabläufen so verhalten, wie im Siemens Nixdorf Styleguide beschrieben.

Dialog Builder

Die Programmierung eines Bediensystems direkt in C unter Verwendung der beschriebenen X Window- und OSF/Motif-Schnittstellen erfordert viel Erfahrung und Fingerspitzengefühl im Umgang mit Bedienoberflächen. Das Design einer Bedienoberfläche ist ein äußerst wichtiger Aspekt der Software, da die Akzeptanz eines Software-Produkts wesentlich vom Design der Bedienoberfläche abhängt. Aus diesem Grund wurde die Entscheidung getroffen, das Werkzeug Dialog Builder als interaktive Hilfe zur Erstellung von SINIX-Bedienoberflächen anzubieten.

Der Dialog Builder basiert auf SINIX/windows Development und kann dazu benutzt werden, OSF/Motif-Bediensysteme zu erstellen, die konform zum Styleguide sind. Mit dem Dialog Builder können sowohl das Layout als auch die Dialogabläufe eines Bediensystems auf einem grafisch interaktiven Weg erstellt werden. Widgets von Siemens Nixdorf können ebenfalls mit dem Dialog Builder bearbeitet werden.

Abbildung 4-4 zeigt die Architektur des Dialog Builder. Der rechte Teil des Abbildung stellt die interaktive Entwicklungsumgebung dar, die auf SINIX und OSF/Motif basiert. Die Laufzeitumgebung für fertiggestellte Bediensysteme ist im linken Teil der Abbildung skizziert. Der Dialog Builder wurde so konzipiert, daß die damit entwickelten Bediensysteme sowohl unter OSF/Motif als auch unter MS-Windows ablauffähig sind.

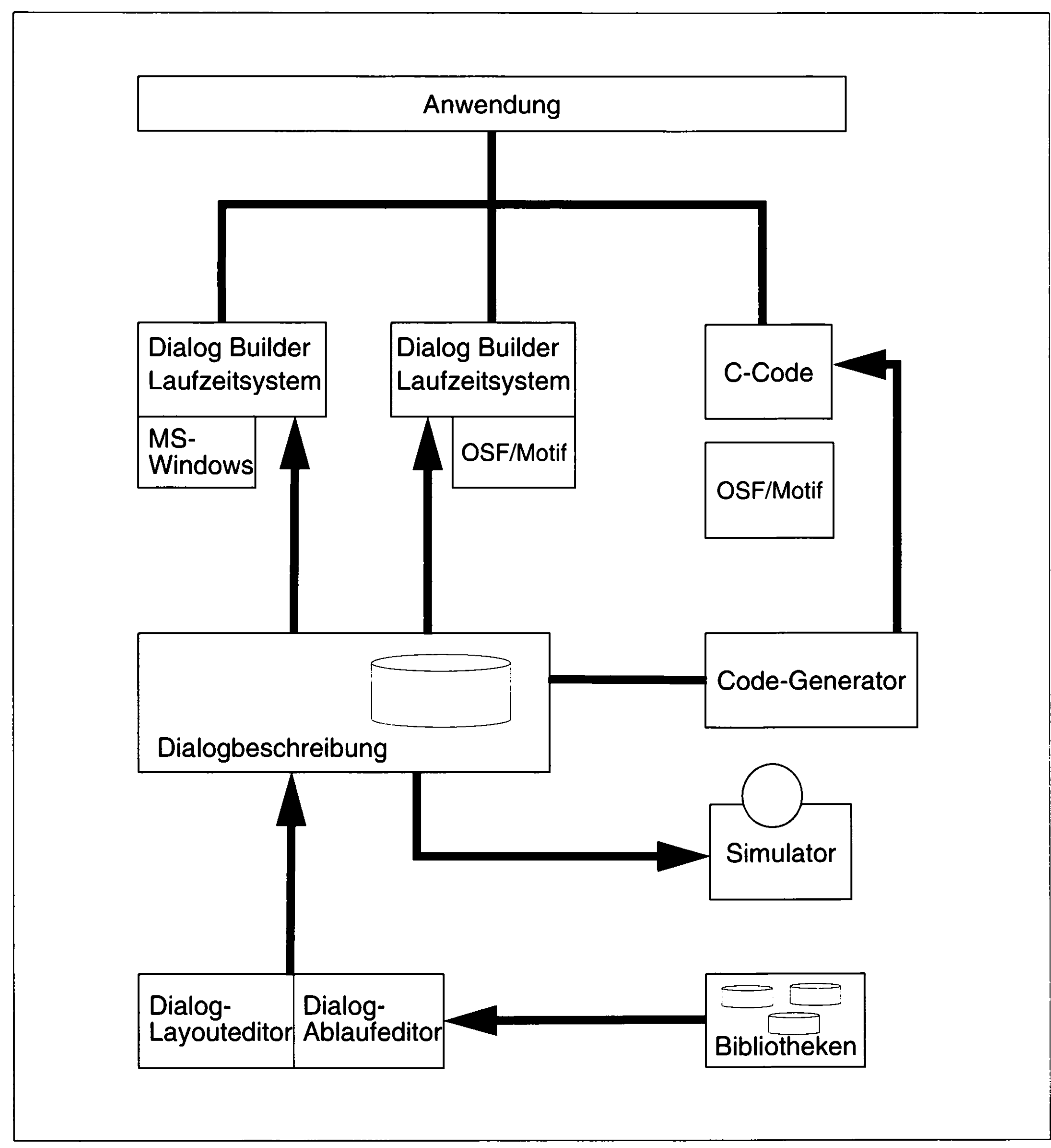


Abbildung 4-4: Der Dialog-Builder

Der grafische Layout-Editor, der nach dem WYSIWYG Prinzip (**W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **G**et) arbeitet, ermöglicht dem Entwickler eine statische Anordnung des Bildschirmlayouts, z.B. der Lage und Größe von Fenstern und Buttons. Das dyna-

mische Verhalten der Bedienoberfläche, wie beispielsweise die Reaktion auf das Aktivieren eines Buttons, wird im Dialogablaufeditor festgelegt. Durch Simulation kann das erstellte Bediensystem zu einem frühen Zeitpunkt getestet werden.

Die Anbindung eines mit dem Dialog Builder erstellten Bediensystems an die eigentliche Anwendung erfolgt entweder über ein Laufzeitsystem oder über die vom Codegenerator erzeugten C-Module. Bei Verwendung des Codegenerators werden die C-Module des Bediensystems und die Module der Anwendung kompiliert und gebunden. Bei Verwendung des Laufzeitsystems werden spezielle Laufzeitmodule mit dem Anwendungscode gebunden. Während die Anwendung abläuft, liest das Laufzeitsystem die Dialogbeschreibung ein, erzeugt die entsprechenden Dialogobjekte und interpretiert die vorher erzeugten Dialogablaufskripts. Wie in Abbildung 4-4 skizziert, steht sowohl für OSF/Motif als auch für MS-Windows ein Laufzeitsystem zur Verfügung.

Wiederverwendbarkeit und Standardisierung von Bedienoberflächen werden im Dialog Builder durch ein Bibliothekskonzept unterstützt. Diese Bibliotheken bestehen aus Sammlungen vorgefertigter Module von Dialogobjekten. Die Bibliotheken können mit Hilfe eines Browsers durchsucht werden. Einzelne Objekte können dann ausgewählt und in das zu erstellende Bediensystem kopiert werden.

Die Bibliothek des Siemens Nixdorf Styleguides enthält Muster für alle Dialogobjekte, die im Siemens Nixdorf Styleguide beschrieben sind. Bei Verwendung dieser Bibliotheken wird die Konformität der erstellten Bedienoberfläche ebenso garantiert wie die Portabilität zwischen OSF/Motif und MS-Windows. Die OSF/Motif-Bibliothek enthält alle von OSF/Motif angebotenen Dialogobjekte. Der Entwickler einer Bedienoberfläche kann bei der Arbeit mit dem Dialog Builder auch auf diese Bibliotheken zugreifen.

Unter Zuhilfenahme der Objekte aus diesen Bibliotheken kann der Entwickler die von ihm gewünschte Bedienoberfläche interaktiv zusammensetzen. Reaktionen auf Eingaben des Anwenders werden mit Hilfe einer Dialogablaufsprache im Dialogablaufeditor definiert.

Bediensysteme, die mit Dialog Builder erstellt wurden, werden in Dialogbeschreibungs-Dateien im ASCII Format abgelegt. Die Dialogbeschreibung enthält eine Layoutbeschreibung und die Dialogsteuerung der Bedienoberfläche. Bedienoberflächen, die mit Hilfe des Dialog Builder entwickelt wurden, können über die NLS-Funktionalität an unterschiedliche landessprachliche Umgebungen angepaßt werden.

Daneben ist im Dialog Builder ein Dialogsimulator enthalten, mit dem der erstellte Dialogablauf ausgeführt und sofort überprüft werden kann. Der Dialogsimulator verfügt über eine Trace-Funktion, die die Überprüfung der erstellten Ablaufskripts wesentlich erleichtert.

Aus den Dialogbeschreibungen kann bei Bedarf auch C-Code erzeugt werden. Dabei können sowohl das Layout als auch die Ablaufsteuerung in den C-Code miteinbezogen werden. Die aus dem C-Code generierbare Anwendung ist auch ohne das Laufzeitsystem des Dialog Builder ablauffähig, so daß eine problemlose Portierung auf andere Plattformen ermöglicht wird.

XVision (MS-Windows)

XVision ermöglicht dem PC-Anwender von MS-Windows 3.1 oder einer neueren MS-Windows-Version, an einem Bildschirm gleichzeitig in der PC-Welt und in der UNIX-Welt zu arbeiten. Realisiert wird dies über eine X-Terminal-Emulation unter MS-Windows. Auch zwischen X Window-Anwendungen und MS-Windows-Anwendungen (sowie zwischen X Window-Anwendungen) ist die Verwendung der Cut-and-Paste-Funktionalität möglich.

Der PC muß über eine LAN-Karte, die als Ethernet Adapter dient, und TCP/IP (LAN1)-Software verfügen. (Dies ist abhängig vom Typ des Netzwerks. Auch die Verwendung eines Tokenring-Adapters in einem Token Ring-Netzwerk wäre möglich.) Folgende zusätzliche Hardwarevoraussetzungen sind erforderlich: Ein PC mit 80386 Prozessor (oder höher) und 4 MB Arbeitsspeicher, eine Maus mit Drei-Tasten-Funktionalität (für OSF/Motif) sowie ein VGA-Grafikadapter. Die verwendete MS-Windows-Version muß MS-Windows 3.1 oder höher sein. Auf dem SINIX-Rechner müssen das X Window System und X-Anwendungen installiert sein.

Im Ein-Fenster-Betrieb hat die X Window-Umgebung ein einziges MS-Windows-Fenster. In diesem einen Fenster entsprechen Erscheinungsbild und Bedienung der X-Umgebung (OSF/Motif, Open Look). Die Steuerung dieses Fensters erfolgt über die Fensterverwalter der X-Umgebung. Im Mehr-Fenster-Betrieb stellt XVision auch eine eigene Fensterverwaltung zur Verfügung, die Erscheinungsbild und Bedienung der Fenster entsprechend dem Standard von MS-Windows-Anwendungen ermöglicht. Die lokale Fensterverwaltung von XVision ist kompatibel zu OSF/Motif.

Der Mehr-Fenster-Betrieb bietet hinsichtlich der Rechenleistung klare Vorteile, da viele Aktionen wie beispielsweise Mausbewegungen lokal verarbeitet werden können, statt über das Netzwerk zur Auswertung durch den Host-Rechner geschickt zu werden. Auf diese Art und Weise werden gleichzeitig der Datenverkehr und die Beanspruchung des Host-Rechners reduziert.

Alphanumerische Bediensysteme

Computerzeitschriften favorisieren schon seit langem Systeme mit grafischen Bedienoberflächen. Alphanumerische Terminals machen jedoch noch immer ein bedeutendes Segment des Terminalmarkts aus. Auch in näherer Zukunft ist hier keine Änderung zu erwarten. Solange die Verwendung eines Terminals auf Aufgaben wie Dateneingabe oder Datenanzeige beschränkt ist, ist die Investition in teure grafikfähige Terminals oft nicht gerechtfertigt.

SINIX wird auch in Zukunft sowohl alphanumerische als auch grafische Bediensysteme unterstützen. Die ersten SINIX-Systeme verfügten über ein leicht verständliches alphanumerisches Menü-Bediensystem für Systemverwaltung, Betrieb des Computers und die Verwendung individueller Anwendungen. Mit dem Übergang zu UNIX SVR4 wurde als Grundlage das alphanumerische Werkzeug zur Entwicklung von Bediensystemen von AT&T, FMLI (**F**orm and **M**enu **L**anguage **I**nterpreter), gewählt. FMLI ist nun im Betriebssystemumfang von SINIX enthalten.

Die Funktionalität des Bediensystems für Anwender und des Bediensystems zur Systemverwaltung wurden aufbauend auf FMLI in die Packages FACE und OA&M (**O**peration, **A**dministration and **M**aintenance) implementiert. Zur Entwicklung alphanumerischer Bediensysteme wird von Siemens Nixdorf ein Satz von Werkzeugen, XMS, angeboten.

FMLI

Bei FMLI (**F**orm and **M**enu **L**anguage **I**nterpreter) handelt es sich um ein Werkzeug, das es dem Anwendungsentwickler ermöglicht, für unterschiedliche Anwendungen alphanumerische Bedienoberflächen zu erstellen. Auf FMLI basierende Bedienoberflächen bestehen meist aus Menüs, Masken und Textfenstern, die der Anwendungsentwickler in einer speziell für diesen Zweck verfügbaren Sprache beschreibt. Diese Sprache, (FML — **F**orm and **M**enu **L**anguage), erlaubt dem Ent-

wickler, Inhalt, Größe, Lage und viele andere Charakteristika der Bildschirmobjekte ebenso wie Aktionen zu programmieren. Diese Aktionen werden durch bestimmte Eingaben des Anwenders ausgelöst. Die Sprache FML verbindet sowohl deskriptive als auch prozedurale Elemente der Programmierung von Bedienoberflächen.

FMLI wertet die Menübeschreibungen, Formatbeschreibungen und Beschreibungen für Textfenster aus und gibt diese auf dem Bildschirm aus. Darüber hinaus steuert FMLI den gesamten Dialogablauf. FMLI leitet die Bewegung innerhalb der Menüs und zwischen verschiedenen Bildschirmobjekten ebenso wie die Erstellung und Positionierung neuer Objekte. FMLI bietet neben der Anzeige von Menüs, Formularen und Textfenstern die Möglichkeit, Ausgaben in einer Meldungszeile anzuzeigen und Kommandos von einer zu diesem Zweck freigehaltenen Kommandozeile einzulesen. Häufig benötigte Funktionen können über zusätzliche Funktionstasten ausgeführt werden.

Der auswertende Ansatz von FMLI eröffnet die Vorteile eines kurzfristigen Prototypeinsatzes und einfacher Wartung. Ab der Version SINIX V5.41 wurde FMLI von Siemens Nixdorf in Zusammenarbeit mit USL internationalisiert, d.h. Anwendungen können in sprachunabhängiger Form kodiert werden. Meldungen und Texte werden zur Laufzeit der Anwendung aus Sprachkatalogen oder speziellen Dateiverzeichnissen eingelesen. Eine Anwendung muß so nur einmal programmiert werden und kann leicht an andere Sprachen angepaßt werden, indem lediglich die Kataloge oder Texte übersetzt werden. Die Internationalisierung von FMLI durch Siemens Nixdorf wurde in SVR4 integriert und ist nun Bestandteil der Hauptlinie von SVR4, das von USL an UNIX-Lizenznehmer geliefert wird.

FACE

FACE (**F**ramed **A**ccess **C**ommand **E**nvironment) ist ein menügesteuertes Bediensystem für SINIX-Systeme. FACE basiert auf FMLI, das Menüs und Bildschirmmasken für alphanumerische Terminals zur Verfügung stellt. Dadurch ist FACE auf allen Terminals ablauffähig und nicht auf grafikfähige Bildschirme beschränkt. Die Originalversion von FACE ist eine Komponente von UNIX SVR4, die von USL geliefert wird. Auch diese FACE-Version wurde 1991 von Siemens Nixdorf in Zusammenarbeit mit USL parallel zu FMLI internationalisiert. FACE 4.2 baut auf dieser internationalisierten Version auf und beinhaltet zusätzliche funktionale Erweiterungen durch Siemens Nixdorf (Postdienste, Unterstützung externer Speichermedien

wie Disketten, Magnetbandkassetten und CD-ROM sowie weitere kleinere Funktionen). Die derzeit gültige Version FACE 4.3 ist in einer deutschen und einer englischen Version erhältlich.

Über diese Bedienoberfläche können sehr unterschiedliche und häufig anfallende Funktionen ausgeführt werden. Der Anwender muß lediglich über geringe Kenntnisse des Betriebssystems SINIX verfügen. Alle Operationen werden von FACE über Fenster ausgeführt. Der Anwender erhält während des Programmablaufs Anweisungen und kann sich bei Bedarf über die aktuelle Funktion oder das aktuelle Fenster einen Hilfetext ausgeben lassen.

Nach dem Starten von FACE erhält der Anwender ein Menü, von dem aus er andere Menüs und Fenster öffnen kann. Diese werden auf dem Bildschirm ähnlich wie unter X Window, OSF/Motif oder MS-Windows in Überlappungstechnik präsentiert. Der Anwender kann unterschiedliche Aufgaben erledigen, indem er zwischen den Fenstern wechselt. Eine simultane Ausführung mehrerer Prozesse in unterschiedlichen Fenstern ist jedoch nicht möglich.

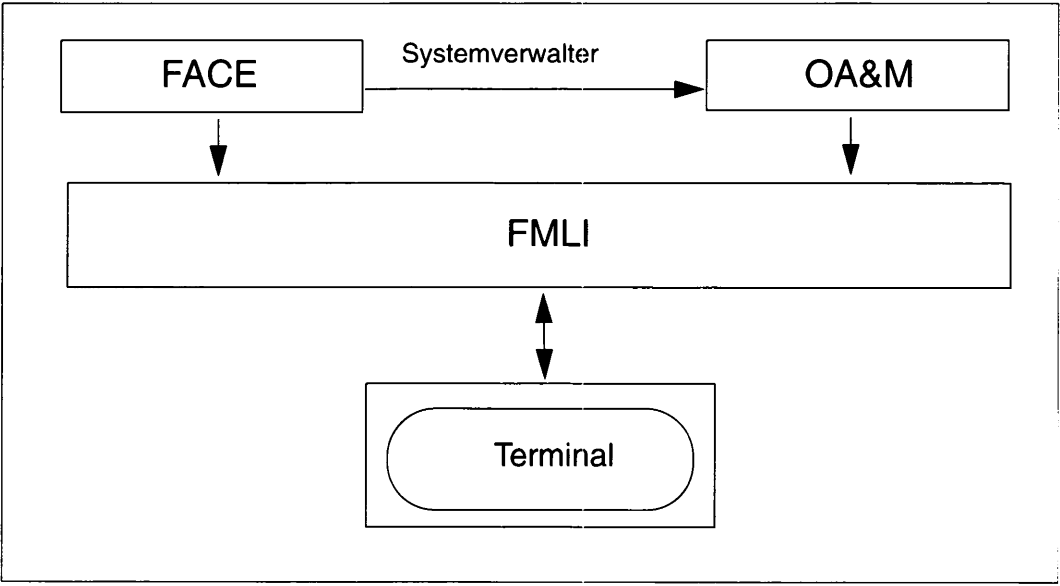


Abbildung 4-5: FACE

XMS

Das strategische Werkzeug von Siemens Nixdorf zur Erstellung alphanumerischer Bediensysteme ist XMS. XMS besteht aus einem reichhaltigen Angebot an Werkzeugen:

Layout-Editor

Der nach dem WYSIWYG-Prinzip arbeitende Layout-Editor unterstützt die Strukturierung von Bildschirmlayouts. Von ihm hängt die Bildschirmdarstellung ab. Zusätzlich zur Geometrie einzelner Bildschirme kann der Entwickler hier auch die Bildschirmattribute (z.B. Formate) und Plausibilitätsregeln für Bildschirmfelder definieren.

Hilfetext-Editor

Hilfe- und Meldungstexte werden unter Verwendung des Hilfetext-Editors erstellt und bestimmten Feldern zugewiesen. Innerhalb dieser Texte können Platzhalter für Ersatzzeichenketten verwendet und Textbausteine mit Attributen versehen werden. Hilfetexte können auch verkettet und als Auswahllisten benutzt werden.

Dialog-Simulation

Mit diesem Simulationswerkzeug können Masken getestet werden, noch bevor das Anwendungsprogramm geschrieben ist. So können verschiedene Masken zu Testabläufen des Bediensystems verkettet werden.

Programmierung

Um die Programmiersprachen C und COBOL unterstützen zu können, verfügt XMS über eine Ein-/Ausgabesprache. Die Kommandos dieser Sprache sind nach SQL-Syntax aufgebaut. Sie sind für beide Programmiersprachen gleich und werden von einem Precompiler, der der jeweiligen Sprache vorgeschaltet ist, in sprachspezifische Aufrufe umgesetzt.

Ablaufbibliotheken

Die XMS Bibliotheken werden mit der Anwendung zusammengebunden, können aber auch als gemeinsam genutzte Bibliothek dynamisch dazugebunden werden. Sie führen Ein- und Ausgabe entsprechend der Formularbeschreibung durch und übernehmen die Behandlung der Hilfetexte und Meldungen.

AlphaWindow

Systeme mit Fenstertechnik werden seit Jahren benutzt, um auf grafikfähigen Bildschirmen eine produktive Benutzerumgebung anzubieten. Standards wie X Window, OSF/Motif oder MS-Windows sind mittlerweile fest etabliert.

Das Ziel, deren Vorteile auch auf alphanumerischen Bildschirmen zu nutzen, konnte bisher leider nicht verwirklicht werden, da kein angemessener Standard verfügbar war, der bei guter Leistung ausreichende Funktionalität bot. Ein Fenstersystem für alphanumerische Bildschirme, das ausschließlich mit Mitteln der Software realisiert ist, muß unweigerlich an eingeschränkter Leistungsfähigkeit scheitern. Alphanumerische Bildschirme werden im allgemeinen über vergleichsweise langsame serielle Leitungen an Host-Systeme angeschlossen. Bei einer reinen Softwarelösung ist der Datenverkehr, der für die Fensterverwaltung notwendig ist, so groß, daß die Grenzen der Übertragungsleistung schon bald erreicht sind.

Durch die Gründung der **D**isplay **I**ndustry **A**ssociation (DIA), eines Zusammenschlusses der wichtigsten Bildschirmhersteller, wurde ein Neuansatz versucht. Seit Januar 1991 versucht die DIA, einen neuen Standard für alphanumerische Bildschirme, die die Fenstertechnik unterstützen, zu definieren. Bereits nach einem Jahr konnte als Ergebnis die Spezifikation des AlphaWindow-Terminals (AW) präsentiert werden. Die ersten Bildschirme, die dieser Spezifikation entsprechen, sind seit Ende 1992 auf dem Markt verfügbar.

Der Aufbau von AlphaWindow sieht vor, daß ein Display Server im Bildschirm alle Funktionen übernimmt, die für die Fensterverwaltung notwendig sind, wie etwa das Begrenzen und Speichern von Fensterinhalten. Ein Fensterverwalter auf dem Host-Rechner stellt die Benutzerschnittstelle zur Verfügung und ermöglicht das Öffnen, Schließen und Verändern von Fenstern. Insoweit entspricht das Konzept dem X Window System. Im Gegensatz zu X wird die Terminalkommunikation mittels

sogenannter virtueller Terminals in der Firmware realisiert. Dadurch wird die Leistungsfähigkeit zusätzlich verbessert. Im AlphaWindow Standard ist die Mausunterstützung ebenfalls integriert.

Die DIA ging bei der Festlegung ihres Standards auf ähnliche Weise wie das X-Konsortium vor. Der Standard definiert lediglich die Bildschirmschnittstelle und die Low-Level Programmierschnittstelle. Bildschirmhersteller können AlphaWindow durch ihre eigenen Erweiterungen ergänzen, z.B. durch Terminal-Emulationen, Softwarehäuser können Fensterverwalter und Toolkits entwickeln und zuliefern.

Die DIA empfiehlt eine AlphaWindow-Architektur, die für die Software-Schnittstelle vier Schichten festlegt:

- Das AlphaWindow-Terminalprotokoll beschreibt die physikalische Terminalschnittstelle (Low-Level Control-Sequenzen).
- Das AlphaWindow-Anwendungsprotokoll legt fest, wie Anwendungen das Terminalprotokoll benutzen sollen, um mit dem Terminalprotokoll und dem Fensterverwalter kommunizieren zu können. Diese Schicht ist die Basis für höhere Software-Schichten, Toolkits, Fensterverwalter und Anwendungen, die eine Low-Level-Schnittstelle benötigen. Das Anwendungsprotokoll stellt eine gemeinsame Grundlage für alle Software-Hersteller dar. Es garantiert, daß die Fensterverwalter und Anwendungen verschiedener Anbieter problemlos nebeneinander auf einem AlphaWindow-Terminal ablaufen können.
- Die Programmierschnittstelle von AlphaWindow, AlphaWindow Application Programming Interface (API), definiert die C-Schnittstelle für das Anwendungsprotokoll. Dieses umfaßt alle Funktionen, die für den Zugriff auf das AlphaWindow-Terminal notwendig sind. Der Entwickler muß sich so nicht mit den Details der Control-Sequenzen befassen. Das API enthält zudem Mechanismen zur Ereignisbehandlung. Es ist zu erwarten, daß Softwarehäuser eine Implementierung des API in Form einer AlphaWindow Bibliothek (AWlib) liefern werden.
- Toolkits anderer Softwarehäuser. In diesem Bereich ist mit der Realisierung von Toolkits durch Softwarehäuser auf Basis der AlphaWindow-Bibliothek AWlib zu rechnen. Die DIA nimmt an, daß sowohl proprietäre Toolkits als auch Standards wie OSF/Motif auf AlphaWindow portiert werden. Dies ist ebenfalls für virtuelle Toolkits wie XVT oder die Toolkits von 4GL zu erwarten.

Das API sollte so beschaffen sein, daß die AlphaWindow-Bibliothek direkt in Anwendungen eingebunden werden kann oder Anwendungen über den Fensterverwalter auf das AlphaWindow-Terminal zugreifen können. Um beiden Anforderungen zu genügen, müßte die AlphaWindow-Bibliothek einen Multiplexer enthalten. Dieser ermöglicht den Ablauf mehrerer Anwendungen auf einem Terminal. Da derzeit eine Arbeitsgruppe der DIA am API arbeitet, sind Änderungen an diesem Konzept absehbar.

Die Zielgruppe für den Einsatz von AlphaWindow bilden Anwender, die die Vorteile des Mehr-Fenster-Betriebs sowie Multi-Tasking und Mausunterstützung nutzen wollen, für eigentliche Grafikanwendungen jedoch keinen Bedarf haben. Der Hauptvorteil von AlphaWindow besteht in der Möglichkeit, unterschiedliche alphanumerische Anwendungen unverändert in einer OSF/Motif-ähnlichen Umgebung ablaufen zu lassen, ohne daß höhere Kosten für X-Terminals und Netzwerke entstehen.

Internationalisierte Benutzerumgebung

Das Betriebssystem SINIX basierte bisher ebenso wie die meisten UNIX-Systeme auf dem ASCII-Zeichensatz und dem amerikanischen Englisch als der Sprache, mit der der Anwender mit dem Computer kommunizieren konnte. Im kommerziellen Bereich mußten natürlich schon interaktive Anwendungen in der Sprache des jeweiligen Anwenders angeboten werden (Sprache umfaßt in diesem Sinn auch nationale und kulturelle Konventionen wie beispielsweise Währungsformate).

Mit großem Aufwand wurden deshalb sogenannte Ländervarianten eines Anwendungsprogrammes produziert. Für jede Sprachvariante waren eine Anpassung der Source, neues Kompilieren usw. erforderlich.

Die zunehmende internationale Verbreitung von UNIX erforderte jedoch eine Erweiterung des Systems, die den länderspezifischen Eigenheiten und den Gewohnheiten der unterschiedlichen Anwender auf flexiblere Weise gerecht wird. Um dieses Ziel zu verwirklichen, begann X/Open 1986 mit der Arbeit an der Definition des NLS (**N**ative **L**anguage **S**ystem).

SINIX-Systeme stellen die notwendigen Mittel (Standard-Internationalisierungsfunktionen, usw.) zur Erstellung internationalisierter Programme zur Verfügung. Zu einem großen Teil handelt es sich schon bei den SINIX-Standard-Kommandos (inklusive der Shell) um internationalisierte Programme.

Zur Beschreibung der SINIX-Programmierwerkzeuge, die die Internationalisierung unterstützen siehe Abschnitt "SINIX API" auf Seite 77.

5 Entwicklungsumgebung

Überblick

Die Entwicklungsumgebung stellt die „Werkstatt" des Programmierers dar, und die verschiedenen Tools sind seine Werkzeuge. Der Programmentwickler benötigt in den Phasen des Entwurfs, der Implementierung und des Testens sehr unterschiedliche Werkzeuge. Während der Entwurfsphase übernehmen die Designwerkzeuge die wichtige Aufgabe, verschiedene Abstraktionsebenen zu verwalten und zu visualisieren, sowie die Beziehung zwischen den Ebenen herzustellen. In der Implementierungsphase verwendet der Programmierer Editoren, um Programme zu entwikkeln und Compiler, um sie in die Maschinensprache zu übersetzen. Über die Programmiersprachen wird dem Programmierer eine über alle SINIX-Systeme einheitliche Schnittstelle (API) zur Verfügung gestellt. Damit er mit den Änderungen auf dem Laufenden bleibt und um die Kontrolle darüber zu behalten, welche Versionen seines Programms kompiliert wurden, braucht der Programmierer ein Konfigurations-Managementsystem. Während der Implementierung und in der Testphase verwendet der Programmierer einen Debugger. Der Debugger versetzt ihn in die Lage, ein Programm zur Ausführungszeit zu überwachen, sei es durch schrittweise Verfolgung oder durch die situationsbedingte Darstellung von Zwischenergebnissen.

Dieses Kapitel behandelt alle genannten Werkzeuge. Zuerst werden die Programmiersprachen, die Compiler und das SINIX API beschrieben. Danach wird kurz auf die Rolle der Editoren, Dienstprogramme, Debugger und des Konfigurations-Managements eingegangen. Das Kapitel endet schließlich mit der Beschreibung einiger Werkzeuge für das computerunterstützte Software-Engineering (Computer-Aided Software Engineering - CASE).

Compiler und Programmiersprachen

Die Evolution der Programmiersprachen in den letzten Jahrzehnten hat zur Schaffung neuer Sprachmittel geführt, die die Strukturierung des Kontrollflusses, abstrakte Datentypen (Pascal), Modularisierung großer Programme (Ada) unterstützen sowie neue Programmierparadigmen, wie die objektorientierte Programmierung (C++) und die deklarative Programmierung (PROLOG) geschaffen haben. Die verschiedenen Programmiersprachen eignen sich für unterschiedliche Zwecke, und das Ziel von Siemens Nixdorf ist es, dem Programmierer eine große Auswahl an Programmiersprachen und erstklassige Compilertechnologien zur Verfügung zu stellen, damit die Programme in den effizientesten ausführbaren Code übersetzt werden können.

Compilertechnologie

Jedes Programm, egal in welcher Programmiersprache geschrieben, muß von einem Compiler (oder einem Interpreter) in eine Folge von Instruktionen übersetzt werden, die vom Prozessor des Rechners ausgeführt werden können. Außerdem muß das übersetzte Programm in einem Format aufbereitet werden, das vom Binder und Lader des Betriebssystems verstanden wird. Das heißt aber auch, daß alle Anwender von Computerprogrammen von der Qualität und der Effektivität der zur Verfügung stehenden Compiler profitieren. Compiler sind spezifisch für den Prozessortyp und das Betriebssystem des jeweiligen Rechners.

Die Prozessortechnologie befindet sich in einer rasanten Entwicklung. Im Abstand von etwa zwei Jahren kommen neue Mikroprozessoren auf den Markt, die höhere Leistungen, enorme Preis-/Leistungsverbesserungen bieten. Dies führt zu einer Vielfalt an Prozessorschnittstellen, die untereinander nicht kompatibel sind. Die Kunden sollen aber auch in der Lage sein, von dieser Entwicklung zu profitieren, ohne ihre Investitionen in Programmentwicklung und -anschaffung zu verlieren. Die SINIX-Compiler garantieren dem Benutzer, daß Programm-Sourcen auf allen SINIX-Systemen ohne Eingriffe in das Programm zum Ablauf gebracht werden können, wenn die im SINIX API definierten Schnittstellen verwendet werden (siehe Abschnitt „SINIX API" auf Seite 75).

Die SINIX-Compiler erfüllen die internationalen Standards für die verschiedenen Programmiersprachen, ein Anspruch, der von autorisierten Validierungsbehörden bestätigt wird. Diese Organisationen haben umfangreiche Prüfprogramme entwikkelt, sogenannte Validation Suites, die nachweisen, daß ein Compiler Programme gemäß den Standards verarbeitet. Compiler, die erfolgreich validiert wurden, sind dadurch von den Validierungsbehörden zertifiziert. Ein Zertifikat muß für jede Prozessorlinie, Betriebssystemlinie und Compilerversion neu erworben werden. Die SINIX-Compiler werden ständig aktuell validiert.

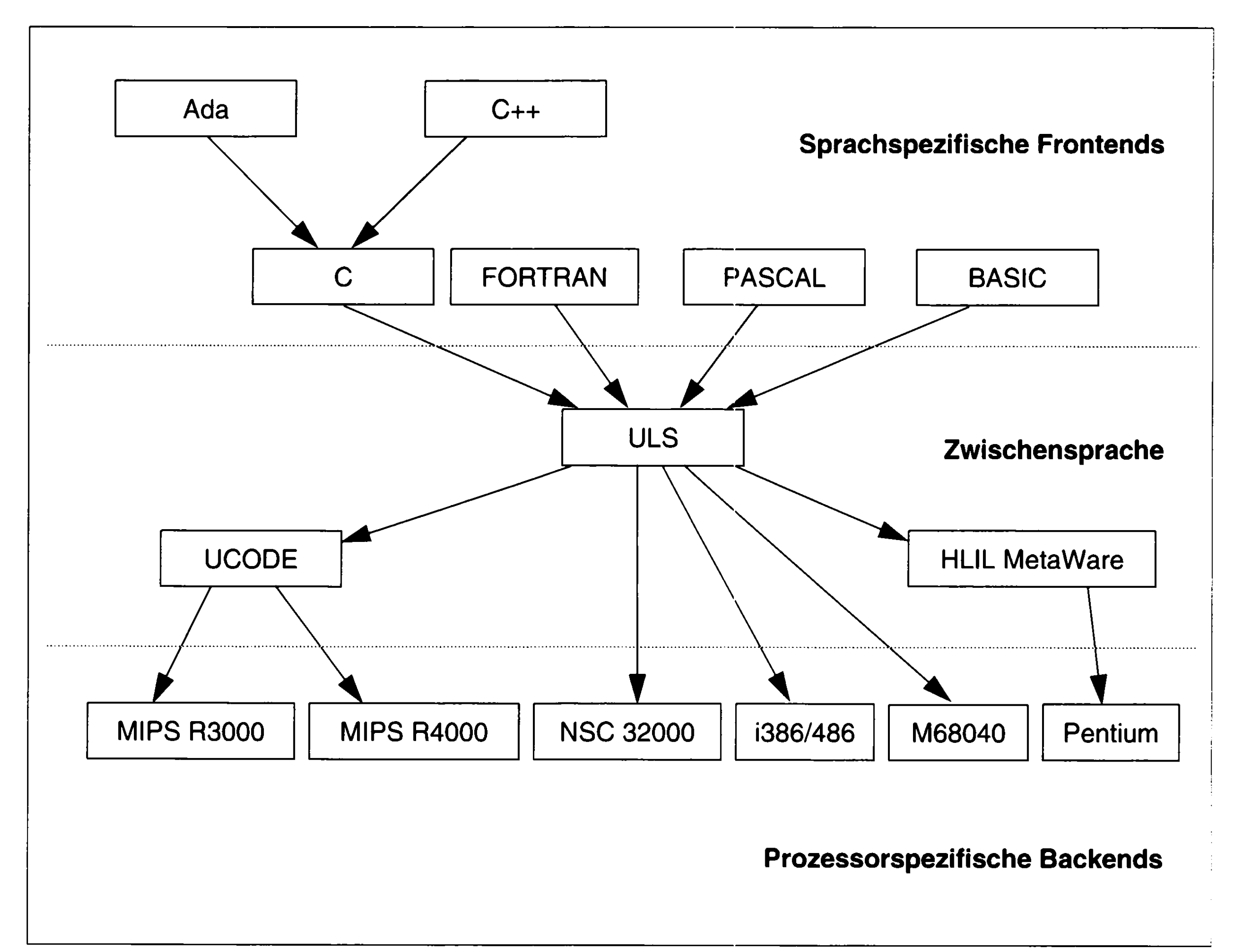


Abbildung 5-1: Die SINIX-Compilerarchitektur verwendet eine gemeinsame Zwischensprache zur Unterstützung einer breiten Palette von Prozessoren und Programmiersprachen.

Compiler bestehen in der Regel aus zwei Teilen, dem Frontend und dem Backend. Das Frontend ist der sprachabhängige Teil, in dem der eingelesene Programmtext analysiert wird. Das Backend stellt den prozessorspezifischen Teil dar, der die vom Prozessor ausführbaren Instruktionsfolgen erzeugt. Ein wichtiger Aspekt der Compilerarchitektur besteht darin, für eine Programmiersprache ein gemeinsames Frontend in den verschiedenen SINIX-Compilern einzusetzen und das Backend für einen speziellen Prozessor so zu entwickeln, damit es für mehrere Frontends, d.h. für mehrere Programmiersprachen, verwendet werden kann. Die Schnittstelle zwischen Frontend und Backend wird Zwischensprache genannt. Die einheitlichen Frontends gewährleisten, daß auf allen SINIX-Systemen ein gemeinsamer Sprachumfang unterstützt wird. Jeder Hersteller von Compilern hat seine eigene Zwischensprachen-Norm. Die Zwischensprachen-Norm der SINIX-Compiler heißt ULS (**U**niversal **L**anguage **S**ystem).

Ada

Die Programmiersprache Ada steht in der Tradition der Algol/Pascal-Sprachfamilie. Sie enthält neben der klassischen Funktionalität anderer Programmiersprachen vor allem Sprachkonstrukte, die ein modernes Software-Engineering unterstützen. Ein Aspekt ist die Unterstützung umfangreicher Programmierprojekte. Ein weiterer Aspekt ist die Unterstützung spezieller Anforderungen aus dem Bereich der Prozeßsteuerungen. Charakteristische Eigenschaften von Ada sind:

- Datenabstraktion
- Modularisierung mit separater Übersetzung und Konsistenzprüfung der Schnittstellen durch den Compiler
- Erzeugung parametrierbarer Softwarekomponenten mit Hilfe von *Generics*
- Exception Handling
- Tasking mit einem Intertask-Kommunikationskonzept, das das *Hoare*-Modell der Communicating Sequential Processes nachbildet

Der Sprachumfang von Ada ist durch die identischen Standards ANSI/MIL-STD 1815A, ISO 8652 und DIN 66268 verbindlich festgelegt.

SINIX-Ada ist eine Portierung des von der Firma Meridian Software Systems lizensierten Compilersystems. Darin sind die Teile Compiler, Optimierer, Binder, Bibliothekssystem und Ada-Debugger enthalten. Neben den Standardpaketen gemäß der Ada-Norm enthält die Bibliothek weitere vordefinierte Pakete, die der Vereinfachung der Ein-/Ausgabe dienen und andere nützliche Funktionen zur Verfügung stellen. Die SINIX-Systemschnittstelle wird mit dem Package Musix unterstützt.

BASIC

SINIX-BASIC ist mit seinem Interpreter ein komfortables Programmiersystem, insbesondere für den weniger routinierten Programmierer. Der Interpreter analysiert während des Editierens die Programmzeile auf Syntaxfehler und während des Programmablaufs auf Laufzeitfehler, bevor sie ausgeführt wird. Das erleichtert die Erstellung und die Korrektur von Programmen. Zusätzliche Hilfen bieten der *Pretty Printer*, der zum Aufzeigen logischer Strukturen dient, und die HELP-Funktion.

BASIC ist heute eine leistungsfähige Programmiersprache. Insgesamt stehen dem Programmierer 170 verschiedene Anweisungen, Funktionen und Kommandos zur Verfügung. Es existieren außerdem mathematische Basisfunktionen und Stringhandling-Funktionen, und der Programmierer kann zwischen Dezimal- und Gleitpunktarithmetik auswählen. Für die Verarbeitung der Daten stehen sequentielle, indizierte und direkte Zugriffsmethoden zur Verfügung.

Um für die fertig ausgetesteten Programme den Interpretations-Overhead zu vermeiden, können BASIC-Programme auch kompiliert werden. Für beide Ausführungsarten des Programms, Interpreter- und Compiler-Programm, gibt es einen Sprachanschluß für C-Objektmodule.

C

Der C-Compiler des Entwicklungspakets unterstützt sowohl den durch ANSI/ISO standardisierten als auch den von Kernighan & Ritchie [B.W. Kernighan & D.M. Ritchie: Programmieren in C, 1983] definierten Sprachumfang. Der Sprachumfang von K&R ist konform zu den X/Open Portability Guides XPG2 und XPG3. Der Sprachumfang von ANSI/ISO entspricht dem Sprachumfang von C im XPG4.

Auf den SINIX-Systemen werden Spracherweiterungen durch den Compiler unterstützt, die vom UNIX-PCC-Compiler definiert wurden (z.B. #pragma weak).

Das Frontend erzeugt den ULS-Zwischencode für den Optimierer bzw. den nachgeschalteten Codegenerator. Es wird auf allen SINIX-Systemen und dem BS2000 eingesetzt. Damit ist die Portabilität der C-Quellen sichergestellt. Die vom Compiler generierten Objekte entsprechen den Anforderungen der SINIX-Systeme hinsichtlich des erzeugten Objektformats ELF und den übrigen Konventionen, die für jede unterstützte Architektur in einem Application Binary Interface dokumentiert sind (z.B. Registersicherung, Calling-Konventionen). Somit können die Objekte ohne Probleme mit denen des UNIX-C-Compilers von USL zu einem ablauffähigen Programm gebunden werden.

Der C-Compiler erzeugt generell gemeinsam benutzbaren Code; gemeinsam benutzte Bibliotheken können ebenfalls mit ihm erstellt werden. Zum Testen der erzeugten Objekte mit dem Debugger DBX wird optional die Symbolinformation im sogenannten DWARF-Format erzeugt, das als De-facto-Standard der ULS gilt.

C++

Der SINIX-C++ V2.1 Compiler unterstützt den Sprachumfang, der in dem Buch von Ellis/Stroustrup [The Annotated C++ Reference Manual, 1990] beschrieben ist. Der Sprachumfang umfaßt insbesondere Einfach- und Mehrfachvererbung, abstrakte Klassen und Polymorphismus. Die Stream I/O entspricht der Definition von AT&T/USL. Mit der Version C++ V3.0 wird das Sprachmittel „Templates" unterstützt. Für die Version C++ V4.0 wird voraussichtlich das „Exception Handling“ angeboten. Eine Standardisierung von C++ durch ANSI und ISO/IEC ist ab 1995 zu erwarten. Eine Klassenbibliothek mit fundamentalen Containerklassen kann zusätzlich bezogen werden.

Der Compiler, ein Sprach-zu-Sprach-Übersetzer, erzeugt aus dem C++-Quellprogramm ein C-Quellprogramm, das dann vom C-Compiler übersetzt wird. Der Sprach-zu-Sprach-Übersetzer und die Bibliotheken wurden von USL lizensiert und auf SINIX portiert.

Mit dem Debugger DBX können die C++-Programme symbolisch getestet werden. Insbesondere werden auch objektorientierte Eigenschaften, wie z.B. Mehrfachvererbung, Memberfunktionen und Operatorfunktionen vom Debugger DBX unterstützt.

Die Objekterzeugung erfolgt vollständig durch den C-Compiler. Damit genügen die generierten Objekte den Anforderungen der SINIX-Systeme hinsichtlich des erzeugten Objektformats ELF und den übrigen Konventionen, die für jede unterstützte Architektur in einem entsprechenden Application Binary Interface dokumentiert sind.

COBOL

Der SINIX-COBOL-Compiler COB85 ist ein von Micro Focus lizensiertes Produkt. Er unterstützt neben dem aktuellen Standard ANS85 auch noch den älteren Standard ANS74, sowie die X/Open Portability Guides XPG2, XPG3 und in Kürze auch XPG4. Außerdem sind IBM-, Ryan McFarland-, Data General- und Microsoft-kompatible Sprachumfänge einstellbar. Damit bietet COB85 einen modernen Sprachumfang mit strukturierter Programmierung, reichhaltiger Verarbeitung numerischer Daten und Zeichenreihen, leistungsfähiger Ein-/Ausgabe-Funktionalität mit Mehrbenutzer-Fähigkeit, integrierter Bildschirmverarbeitung, Report Writer und Sort. COB85 ist in die Systemumgebung der SINIX-Systeme integriert und unterstützt gemeinsam benutzte Bibliotheken für das Laufzeitsystem und die COBOL-Objekte.

Der COBOL-Compiler COB85 besteht aus drei Teilen:

- Der Checker überprüft das COBOL-Programm syntaktisch und semantisch und erzeugt einen Zwischencode (int-code), der interpretierbar ist. Dieser Zwischencode kann mit dem Debugger ANIM85 getestet werden.
- Der Native Code Generator erzeugt aus dem Zwischencode einen ausführbaren Maschinencode, der gebunden und ausgeführt werden kann.
- Das Laufzeitsystem wird zu dem ausführbaren Maschinencode dazugebunden, um den korrekten Ablauf des Programms zu gewährleisten. Daneben enthält das Laufzeitsystem den Interpreter für den Zwischencode.

Damit hat der Programmierer die Möglichkeit, seinen Entwicklungszyklus zu optimieren, ohne Nachteile für einen optimalen Ablauf seiner Programme in Kauf nehmen zu müssen.

Das COBOL-Compilersystem bietet eine umfangreiche Werkzeugumgebung:

- **ANIM85** ist ein Debugger, der ein bildschirmorientiertes Testen von COB85-Programmen auf Sourceebene gestattet. Er erlaubt die Unterbrechung von Programmen an beliebigen Anweisungen, die Ablaufverfolgung und das Verändern von Variablen.
- **FORMS2** ist ein Hilfsmittel zur Entwicklung von Masken auf der Basis des XPG2 Screen Handlings.
- **Advanced Environment** ist ein Zusatzprodukt, das Werkzeuge anbietet, die die Entwicklung von COBOL-Programmen produktiver gestalten:
  - Advanced Animator erlaubt, zusätzlich zu den normalen Animator-Funktionen, die Ausgabe von Programmstrukturen, den Test der Programmstrukturen und die Überwachung der Variablen in Fenstern. Außerdem ist ein Analyzer zur Analyse der COBOL-Programme enthalten.
  - Screens bietet die komfortable, interaktive Erzeugung von Masken auf Basis des XPG3 Screen Handlings.
  - Session Recorder unterstützt die automatische Aufzeichnung des Programmablaufs und dessen Wiederholung.

Die nächste Version von COB85 wird auf Micro Focus COBOL V3 basieren. Sie wird die Definition des X/Open XPG4 voll unterstützen. Außerdem werden Intrinsic Functions, ein offizielles Addendum zum ANS85-Standard, angeboten. Dieses Addendum enthält Funktionen für die technisch-wissenschaftliche und kommerzielle Datenverarbeitung.

Die Werkzeugumgebung ist durch eine komfortable Menüoberfläche handlich gemacht worden. Daneben wird als zusätzliches Produkt das Dialog System zur Maskenentwicklung angeboten. Dieses Werkzeug erlaubt es dem Programmierer, die Maskendefinition vollkommen aus einem Programm auszulagern und dadurch mit einem Programm verschiedene Bildschirmmasken zu bedienen, ohne neu übersetzen zu müssen.

FORTRAN

FORTRAN ist die wichtigste Programmiersprache für technische und wissenschaftliche Anwendungen. Die heute noch im wesentlichen verwendete Sprache basiert auf dem FORTRAN 77-Standard gemäß der Norm ANSI X3.9-1978.

Der FORTRAN77-Compiler für SINIX ist eine Eigenentwicklung von Siemens Nixdorf. Er enthält einige Erweiterungen zum Sprachstandard, wie z.B. zusätzliche Datentypen und die Verwendung des C-Präprozessors. Der Compiler ist auf allen SINIX-Systemen validiert und mit der optimierten mathematischen Bibliothek XPG3- und XPG4-konform.

Der Compiler erzeugt mehrfach benutzbare Objekte. Mehrfach benutzbare Funktionen können dynamisch nachgeladen werden. Für den Programmtest steht die symbolische Testhilfe DBX zur Verfügung.

Ein Compiler für den FORTRAN90-Standard ist in der Entwicklung. Er bietet leistungsfähige Komponenten zur Bearbeitung von Feldern, benutzerdefinierte Datentypen, Pointer und ein leistungsfähiges Modulkonzept. Der FORTRAN90-Compiler von Siemens Nixdorf wird die meisten Spracherweiterungen des BS2000-FORTRAN-Compilers FOR1 unterstützen (z.B. die variable Länge von Zeichenketten), um eine einfache Migration zwischen den eigenen Mainframe- und Midrangesystemen zu gewährleisten.

Pascal-XT

Die von N. Wirth entwickelte Programmiersprache Pascal zeichnet sich durch ein besonders klares und kompaktes Sprachdesign aus. Pascal ist vor allem im universitären Bereich verbreitet. Es gibt zahlreiche Compiler, die auf der ISO-Norm 7185 basieren und in der Regel auch Spracherweiterungen unterstützen. Seit 1991 gibt es einen neuen Standard: Extended Pascal ISO 10206. Diese Programmiersprache hat aber noch keine Verbreitung gefunden.

Der Pascal-XT SINIX-Compiler unterstützt ebenfalls eine eigendefinierte Erweiterung des Standards ISO 7185, die zwar syntaktisch abweicht, aber funktional dem Extended Pascal stark verwandt ist. Es gibt einen Programmkonverter, der Pascal-XT in Extended Pascal umwandelt.

Die symbolische Testhilfe PATH ist Teil des Pascal-XT-Compilers. Das gesamte Pascal-XT-System ist eine Eigenentwicklung von Siemens Nixdorf, die auf SINIX und BS2000 verfügbar ist.

PROLOG

Mit der Programmiersprache PROLOG kann eine besonders hohe Produktivität bei der Entwicklung von Anwendungen erreicht werden, die Suchstrategien, Regelmengen, Ableitungs- und Induktionstechniken, symbolisches Rechnen oder Musterverarbeitung zum Inhalt haben. Siemens Nixdorf setzt PROLOG erfolgreich in technologischen Kernbereichen der Produktion und als Implementierungssprache für wissensbasierte Produkte ein.

Mit SNI-PROLOG V3.0 bietet Siemens Nixdorf als erster Hersteller eine zum ISO-Normenentwurf kompatible Implementierung der Programmiersprache PROLOG an.

Darüber hinaus bietet SNI-PROLOG ein ausgereiftes Modul-Konzept, programmierte Behandlung von Ereignissen, Ausnahmen und Signalen, sowie einen optimierenden inkrementellen Compiler und Decompiler. Das Modul-System von SNI-PROLOG wurde als Basis des ISO-Normenentwurfs Teil 2 ausgewählt.

SNI-PROLOG verfügt über eine OSF/Motif-basierte grafische Entwicklungsumgebung mit einem integrierten Editor, einer ausgereiften Testhilfe und einem Hypertext-basierten Hilfesystem. Außerdem gibt es eine Reihe von konfigurierbaren Schnittstellen zu anderen Systemen, wie z.B. eine Datenbankschnittstelle zu INFORMIX. SNI-PROLOG ist nicht nur auf allen SINIX-Plattformen, sondern auch auf BS2000, MS-DOS und anderen UNIX-Plattformen (SGI, SUN, HP, R6000) verfügbar.

Die ausgezeichnete Leistungsfähigkeit der mit SNI-PROLOG erstellten Anwendungen wird durch Optimierungsstrategien wie Klauselindizierung und flaches Backtracking im hochoptimierten inkrementellen Compiler erreicht. Die volle automatische Garbage Collection über alle dynamische Speicherbereiche, sowie die implizite Anpassung des zugewiesenen Speichers an die Bedürfnisse einer Anwendung, führen zu einem ökonomischen Umgang mit Ressourcen bei Ein- und Mehrplatzsystemen.

Zur Optimierung von Problemlösungen haben sich Constraints (Wertebedingungen) als geeignetes Mittel in PROLOG erwiesen. Die Constraint-Technologie von SNI-PROLOG (ab der Version 3.1) bietet mit verschiedenen Klassen von Wertebedingungen neue, wirkungsvolle Mittel zur Lösung von Aufgaben aus Industrie und Operations Research an, die mit traditionellen PROLOG-Systemen oder konventioneller Software nur schwer oder überhaupt nicht zu behandeln sind. Beispiele dafür sind die dynamische Aufteilung von Ressourcen oder flexibles Scheduling als Anwendung von numerischen Constraints, sowie die Verifikation komplexer Systeme mit Boolschen Constraints.

SINIX API

Die Bedeutung von Schnittstellen für die Anwendungsprogrammierung hat mit der Verpflichtung zu offenen Systemen in den vergangenen Jahren zugenommen. Ein API (**A**pplication **P**rogramming **I**nterface) ist eine offiziell unterstützte und veröffentlichte Definition einer Programmierplattform. Aber erst mit der Gewährleistung von APIs hat der Kunde zusätzlich die Sicherheit, daß Anwendungen, die unter Verwendung dieser Schnittstellen erstellt wurden, mit der nächsten Ausgabe der Software weiterhin ablauffähig sind. Das SINIX API wird in diesem Sinne von Siemens Nixdorf garantiert, andererseits aber auch regelmäßig um neue Standards erweitert.

Das SINIX API definiert ein offenes System auf der Basis des Betriebssystems System V Release 4 von USL. Es erfüllt die Schnittstellenstandards POSIX, XPG3 und XPG4. Der Kunde ist damit in der Lage, seine Systementscheidungen flexibel zu treffen, ohne Rücksicht auf den Umfang seiner Investitionen in bestehenden Anwendungen, die für SINIX entwickelt wurden, ganz im Gegensatz zu APIs proprietärer Betriebssysteme, die die Auswahl des Kunden beschränken. Um dem Kunden immer die aktuellsten Systemlösungen zur Verfügung zu stellen, verfolgt Siemens Nixdorf diese Standards und folgt auch dem Entwicklungspfad der USL bezüglich der Zukunft des System V Release 4 (SVR4) API.

Im Zentrum des SINIX API steht ein API, das XPG3-konform und künftig auch XPG4-konform ist. Dies ist durch die Erlangung und laufende Erneuerung des XPG3 PLUS-Brandings für SINIX dokumentiert. In anderen Bereichen, wie bei den grafischen Bedienoberflächen (GUI) und den Netzwerkschnittstellen, bestand die Notwendigkeit über den Kern des XPG3-API hinauszugehen. Der Ansatz von Siemens Nixdorf geht davon aus, soweit möglich Industriestandards in das SINIX

API zu integrieren und dort wo es notwendig war, zusätzlich Siemens Nixdorf-spezifische Funktionalität hinzuzufügen. Sobald neue Standards entwickelt werden, fügt Siemens Nixdorf diese in das SINIX API ein und ersetzt gegebenenfalls die Siemens-Nixdorf-spezifischen Schnittstellen durch die Standards. Dies gilt insbesondere für alle Komponenten des XPG4.

Das SINIX API besteht aus den Elementen, die in Abbildung 5-2 aufgeführt sind.

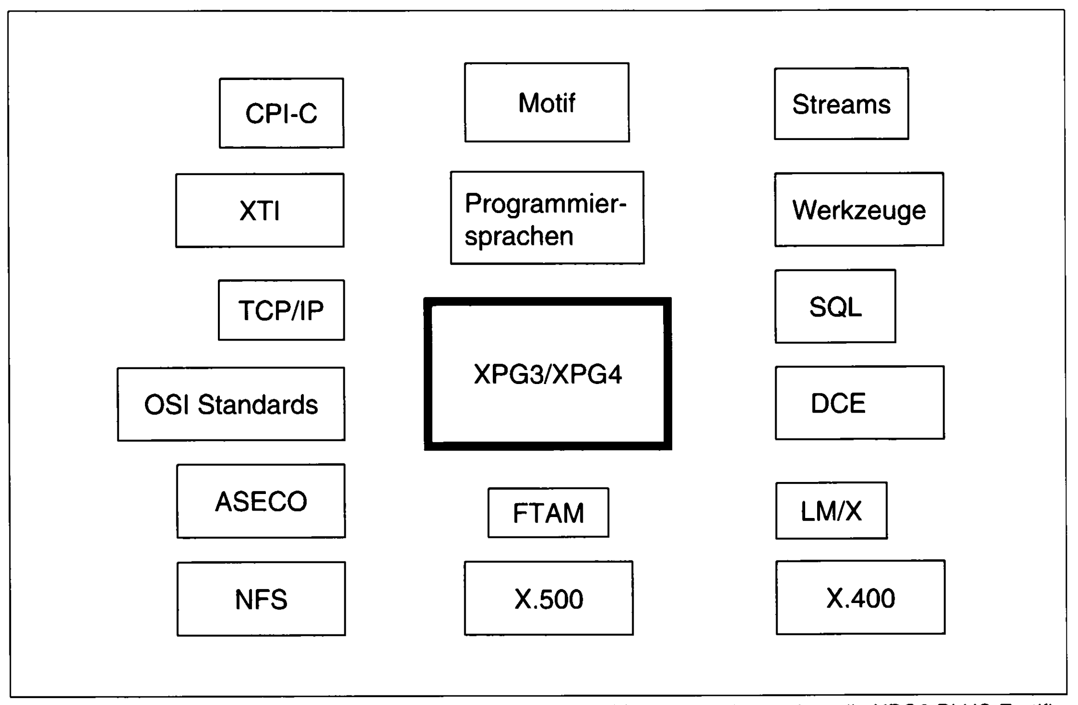


Abbildung 5-2: Im SINIX API sind viele Standards enthalten. Im Mittelpunkt aber stehen die XPG3 PLUS-Zertifizierung und umfassende XPG4-Zertifizierung.

Internationalisierung und Lokalisierung

Die SINIX-Implementierung des NLS (**N**ative **L**anguage **S**ystem) von X/Open ermöglicht die Entwicklung von Anwendungsprogrammen, die mit dem Anwender in den verschiedenen natürlichen Sprachen kommunizieren und die über Informationen länderspezifischer Konventionen verfügen. Solche Programme machen keine Annahmen über die Sprache des Anwenders und den verwendeten Zeichensatz. Die sprach- und kulturspezifischen Daten und Texte werden getrennt von der Progammierlogik gehalten. Man bezeichnet diese Programme als *internationalisierte Programme*.

NLS ermöglicht außerdem, internationalisierte Anwendungsprogramme zur Laufzeit mit der richtigen Landessprache und den länderspezifischen Konventionen zu versorgen. Dieser Prozeß wird als *Lokalisierung* bezeichnet.

Locale

Die länder- und sprachspezifischen Informationen werden als Elemente einer sog. *Locale* zusammengefaßt. Dabei beschreibt die Locale in diesem Fall die vom Anwender einstellbare Arbeitsumgebung, ähnlich der von der Shell bekannten Arbeitsumgebung. Zu einer Locale werden die Beschreibungen der in einem Land oder in einer Region gebräuchlichen Konventionen oder auch die für einen Anwender individuell bevorzugte Arbeitsumgebung zusammengefaßt. Eine Locale darf aber nicht mit einer Sprache verwechselt werden. Sie beschreibt eine wesentlich umfassendere Umgebung.

Entspricht der Name einer Locale den Konventionen aus dem X/Open Portability Guide (Issue 3 oder 4), wird er mit einer Kombination aus den drei Elementen Sprache, Territorium und Zeichensatz benannt. Eine große Anzahl von Localen wird von SINIX-Systemen unterstützt: deutsche, amerikanische, englische, dänische, französische, russische, usw.

Kategorien

Eine Locale läßt sich in noch feinere Gruppen unterteilen, die als *Kategorien* bezeichnet werden. Sie erlauben einen flexibleren Zugriff auf bestimmte kulturelle Informationen. Folgende Kategorien stehen zur Verfügung:

Kategorie	**Einfluß auf:**
LC_ALL	Gesamte internationale Umgebung
LC_CTYPE	Zeichenklassen und Umwandlung von Klein- in Großbuchstaben und umgekehrt
LC_COLLATE	Sortierreihenfolge
LC_TIME	Datums- und Zeitangaben
LC_MONETARY	Währungszeichen und Währungsformat
LC_NUMERIC	Dezimalpunktdarstellung, Exponentzeichen und Tausendertrennzeichen
LC_MESSAGES	Meldungstexte und Antworten auf ja/nein-Fragen

Basismechanismen

Grundsätzlich können zwei Arten von Informationen in einer Locale enthalten sein:

- programmspezifische Informationen: Das sind die Texte, die das Programm ausgibt. Sie werden in sog. *Message-Files* oder *Meldungskatalogen* zusammengefaßt.
- programmunabhängige Informationen: Das wäre z.B. das aktuelle Datum oder die Landeswährung. Diese Information ist vernünftigerweise nur einmal pro Sprache als Locale vorhanden. Die programmunabhängigen Informationen werden in den sog. *NLS-Datenbasen* gehalten. Diese Datenbasen dürfen nicht mit kommerziellen Datenbanken wie INFORMIX verwechselt werden, sie stellen nur eine Sammlung von bestimmten länderspezifischen Daten dar.

Ein internationalisiertes Programm greift erst zur Ablaufzeit auf den Meldungskatalog bzw. die NLS-Datenbasis zu.

Die Einstellung einer den individuellen Erfordernissen entsprechenden nationalisierten Arbeitsumgebung ist mit Hilfe von Umgebungsvariablen möglich. Internationalisierte Programme entnehmen dem Wert der Variablen die Information, welche Locale zu verwenden ist. Auf Informationen bestimmter Kategorien wird gezielt mit Hilfe der entsprechenden Umgebungsvariablen zugegriffen: LC_COLLATE, LC_CTYPE, usw.

Zusätzlich sind folgende Variablen verfügbar:

Die Variable LANG dient als Standardwert für alle Kategorien, die nicht explizit gesetzt sind.

Die Variable LC_ALL setzt die gesamte Umgebung auf die entsprechende Sprache.

Editoren

Editoren sind die Schreibwerkzeuge des Software-Entwicklers, mit denen er Sourcecode erstellt und seine Arbeit dokumentiert. Ohne Editor wäre die moderne Programmentwicklung unmöglich und gerade in diesem Bereich ist die Auswahl des richtigen Werkzeugs entscheidend für Produktivität und Qualität. Die Produktivität des Programmierers kann durch Editoren mit leistungsfähigen Editier- und Suchfunktionen, mit kontextsensitiven Repräsentationshilfen und durch eine effiziente Zusammenarbeit mit Compilern bei der syntaktischen und semantischen Überprüfung von Programmen wesentlich verbessert werden. Ein Editor mit einer klaren, gut strukturierten Kommandoschnittstelle, der mit wenigen Tastenkombinationen bedient werden kann, ist sehr hilfreich. Außerdem spielen grafische und objektorientierte Schnittstellen für Editoren eine zunehmend größere Rolle.

Im SINIX-System gibt es mehrere Editoren. Sie sind für verschiedene Zwecke und Editierumgebungen vorgesehen. Die zeilenorientierten Editoren *ed*, *ex* und *edit* eignen sich für kleinere, schnell auszuführende Operationen. Der stapelorientierte Editor *sed* wird innerhalb von Shellskripts zur Datenbearbeitung verwendet. Die bildschirmorientierten Editoren *vi* und *ced* (SINIX-spezifisch) entsprechen eher dem Bild eines modernen Editors. Während der *vi* durch seine Vielfalt und Allgegenwärtigkeit in der UNIX-Welt überzeugt, macht die einfache Bedienung den *ced* ebenfalls sehr attraktiv.

Der MAXed ist ein SINIX-spezifischer Editor, der eine universell einsetzbare Editierumgebung zur Verfügung stellt, die auf alle Aspekte der Softwareentwicklung abgestimmt ist. Der MAXed bietet bei nur kurzer Einarbeitungszeit und einfacher Bedienung über leicht zu merkende Kommandos sowohl dem Neuling als auch dem Profi einen enormen Funktionsumfang. Er ermöglicht die Erstellung von Sourcecode und der dazugehörigen Dokumentation, ohne daß zusätzlich auf ein Textverarbeitungssystem zurückgegriffen werden muß.

Zu den herausragenden Eigenschaften des MAXed zählen:

- die einfache Bedienung mit konfigurierbaren Funktionstasten
- die kurze Einarbeitungsphase
- die vollständigen und flexiblen Editierfunktionen
- ein bequemer und effizienter Editier-Compiler-Zyklus
- die Unterstützung der Textformatierung
- die garantierte Qualität und Kompatibilität

UNIX-Dienstprogramme

In allen UNIX-Systemen, so auch in SINIX, gibt es zahlreiche Dienstprogramme, die für die Programmentwicklung außerordentlich hilfreich sind. Dies ist ein angenehmer Nebeneffekt der Fortschritte und Verbesserungen, die über die Jahre hinweg durch die Verwendung von UNIX in Universitäten und Forschungseinrichtungen eingeflossen sind. Die drei Beispiele *make, lex* und *yacc* sollen stellvertretend einen Eindruck von diesen Werkzeugen vermitteln.

Das Dienstprogramm *make* ist ein ausgesprochen wertvolles Werkzeug bei der Erstellung und Pflege von Programmen und Programmsystemen. Für Programmierer, die damit Erfahrungen gesammelt haben, ist es überhaupt nicht mehr wegzudenken. Das verwendete Prinzip ist denkbar einfach. Der Programmierer verwendet eine Datei, das sog. „*Makefile*", um zu beschreiben, wie ein bestimmtes Programmsystem zu erzeugen ist. Diese Datei enthält Informationen wie die Liste alle Sourcecode-Dateien, mit welchem Kommando die einzelnen Sourcecode-Dateien übersetzt werden sollen, usw. Der Programmierer braucht dann nur noch das Dienstprogramm *make* mit dem *Makefile* als Argument aufzurufen. *make* liest die Informationen in der Datei *Makefile*, erzeugt die entsprechende Umgebung und

führt die entsprechenden Kommandos zur Übersetzung der Sourcecode-Dateien aus. Wenn die kompilierte Version einer bestimmten Datei bereits die letzten Änderungen enthält, wird die Datei nicht neu übersetzt. Ein Dienstprogramm wie *make* ist selbst bei der Entwicklung einfacher Programme sehr hilfreich. Wenn aber mehrere Programmierer zusammen an einem großen Projekt arbeiten, ist es unverzichtbar. Es stellt sicher, daß keine Programmdatei vergessen wird und alte Dateiversionen nicht versehentlich wiederverwendet werden.

Das Dienstprogramm *lex* führt eine lexikalische Analyse von Textdateien durch. Der Entwickler erzeugt eine Eingabedatei für *lex*, die Zeichenketten und reguläre Ausdrücke enthält, nach denen gesucht werden soll, und schließlich den C-Code, der ausgeführt werden soll, wenn die Zeichenkette oder der reguläre Ausdruck gefunden wurde. Aus der Eingabedatei wird ein C-Programm erzeugt. Dieses Programm kann dann übersetzt und dazu verwendet werden, Textdateien zu durchsuchen und die entsprechende Ausgabe zu erzeugen. Zur Veranschaulichung kann man sich einen Programmierer vorstellen, der in Hunderten von C-Programmen an den verschiedensten Stellen den Namen seiner Firma einprogrammiert hat (Copyright-Vermerke, Bildschirmausgaben, usw.). Nun muß die Firma aufgrund eines Zusammenschlusses ihren Namen ändern. Mit dem Dienstprogramm *lex* kann der Programmierer sehr schnell ein *lex*-Programm schreiben, das den Namen der alten Firma durch den neuen ersetzt. Was zunächst als enorme Anstrengung erschien, ist zu einer einfachen Aufgabe geworden.

Die hintergründig-humorvolle Abkürzung *yacc* steht für „yet another compiler compiler". Dieser Compiler wird dazu verwendet, Programme zu erzeugen, die Eingabedateien analysieren können, welche in einer speziellen, möglicherweise vom Entwickler erfundenen „Programmiersprache" geschrieben sind. Der Entwickler erzeugt eine Eingabedatei mit den Regeln dieser „Programmiersprache" und benutzt *lex* dann zur Generierung des Programms, das die Analyse durchführt. Eine mögliche Anwendung für *yacc* wäre die Definition einer kleinen Programmiersprache, die es Entwicklern erlaubt, Programme für ganz spezifische Aufgabentypen zu erzeugen. Anschließend könnten sie das mit *yacc* generierte Programm dazu verwenden, um daraus einen C-Code zu erzeugen.

Debugger

Ein Debugger ist ein Werkzeug zur Analyse des dynamischen Verhaltens einer Anwendung. Unter der Kontrolle eines Debuggers kann der konkrete Ablauf einer Anwendung verfolgt werden. Außerdem ist es möglich, in den Programmablauf einzugreifen, indem z.B. Variablen neue Werte zugewiesen werden. Debugger werden häufig eingesetzt, um Laufzeitfehler zu finden, die statisch, etwa durch einen Compiler, nicht erkannt werden können. Die Bezeichnung Debugger ist etwas irreführend, denn der Debugger behebt selbst keine Fehler, sondern hilft dem Entwickler, sie zu lokalisieren. Neben der Fehlersuche wird der Debugger auch bei der Einarbeitung eines Entwicklers in fremden Code und bei der kritischen Beobachtung von funktionierenden Datenstrukturen oder Algorithmen hinsichtlich ihrer Performanz oder ihres Ressourcenverbrauchs verwendet.

In Entwicklungsumgebungen mit grafischen Workstations werden Fenstertechniken, grafische Repräsentation und Interaktionsmechnismen für eine simultane Wiedergabe des Prozeßstatus, des Quellprogrammkontextes und der Bildschirmausgaben des Prozesses mit dem Debugger verwendet.

Die Siemens Nixdorf Sprachen-Systeme COBOL, Pascal, PROLOG und Ada verfügen über jeweils eigene Debugger. C, C++ und FORTRAN77 haben einen gemeinsamen Debugger: DBX. Der Debugger DBX wurde ursprünglich an der University of California in Berkeley entwickelt. DBX hat in der UNIX-Welt eine weite Verbreitung gefunden, was dazu geführt hat, daß viele Entwickler mit seiner Basisfunktionalität vertraut sind.

Die Siemens-Nixdorf-Version des DBX ist auf allen Systemen mit SINIX-Compilern verfügbar. API-konforme Anwendungen werden vom DBX voll unterstützt. DBX kann auch auf der Maschinenebene arbeiten. Außerdem unterstützt DBX die übliche Debugger-Funktionalität wie Step, Trace und das Setzen von bedingten und unbedingten Haltepunkten. Variablen können automatisch überwacht werden. Die symbolische Adressierung von Variablen und Funktionen ist möglich. Einzelne Funktionen aus der Anwendung können direkt vom DBX aufgerufen werden. Diese Funktionalität ist bei modularem Testen sehr hilfreich. Der DBX-Benutzer kann beliebige Ausdrücke, bestehend aus Programm-, Variablen- und Funktionsaufrufen von DBX, evaluieren lassen. Darüberhinaus besitzt der Debugger DBX eine Vielzahl weiterer Leistungen, die in bestimmten Situationen nützlich sind.

Die Basisfunktionalität ist bei allen drei unterstützten Programmiersprachen gleich. Darüber hinaus unterstützt der DBX die Besonderheiten der einzelnen Programmiersprachen. In C++ werden u.a. überladene Funktionen und Operatoren unterstützt. Der Anwender arbeitet direkt mit den symbolischen Namen aus der Anwendung und nicht mit den künstlichen Namen, die vom Compiler erzeugt werden. Bei FORTRAN-Programmen werden u.a. COMMON, EQUIVALENCE und Entry-Points unterstützt.

Der Debugger DBX ist auch in der Lage, Anwendungen zu verarbeiten, die aus Modulen unterschiedlicher Source-Programmiersprachen bestehen.

Die neuste Version von DBX wurde um eine OSF/Motif-basierte grafische Bedienoberfläche erweitert. Dadurch wird nicht nur die Bedienung erleichtert, sondern die Darstellung der Anwendung und ihrer Daten wird wesentlich übersichtlicher, wodurch Geschwindigkeit und Zuverlässigkeit der Entwicklungsarbeit zunehmen. Diese Version läßt sich auch in die weiter unten auf Seite 92 beschriebene OSCE-Entwicklungsumgebung einbetten. Die Fähigkeit, Mehr-Prozeß-Anwendungen und gemeinsam benutzte Bibliotheken (shared libraries) zu analysieren, versetzt den Debugger DBX in die Lage, den Anforderungen von Entwicklern komplexer Anwendungen gerecht zu werden.

Konfigurations-Management

In vielen Projekten wird heutzutage das Software-Konfigurations-Management noch immer nicht konsequent eingesetzt. Dabei ist es eine Disziplin, die ähnlich der Qualitätssicherung wesentliche Beiträge zum Erfolg eines Projekts leisten kann. Während man bei der Hardware eine Änderung oftmals noch sehen kann, ist dies bei der Software häufig nicht der Fall. Deshalb können Projektstand und Fortschritte nur sehr schwer transparent gemacht werden. Es ist kompliziert, die Softwareänderungen zu verfolgen und Nebeneffekte sowie Interaktionen zu überwachen. Selbst die kleinsten Manipulationen am Code können große Probleme verursachen. Ohne gesicherte Informationen über Änderungen ist es oft schwer, die Änderungen zu entdecken, die zu einem Problem geführt haben. Der Änderungsprozeß in der Softwareentwicklung muß durch ein Konfigurations-Management kontrolliert und verfolgt werden, wenn ein Projekt effizient arbeiten und Erfolg haben soll. Um dieses Ziel zu erreichen müssen folgende Aufgaben erfüllt werden:

- eindeutige Identifikation aller Bestandteile einer Software (configuration identification)
- Einbringen und Freigeben von Änderungen (configuration control)
- Verfolgen und Sichtbarmachen von Änderungen (configuration auditing)
- objektives Berichten über den Änderungsstand an das Management (configuration accounting).

Sollen diese Aufgaben konsistent und zuverlässig durchgeführt werden, empfiehlt sich der Einsatz eines rechnergestützten Verfahrens.

Unter UNIX stehen zwei Pakete von Programmen zur Verfügung, die die Verwaltung von Software erlauben. Das Source Code Control System (*sccs*), das in SINIX enthalten und im X/Open Portability Guide (XPG) beschrieben ist, sowie das Revision Control System (*rcs*), das als Public Domain Software erhältlich ist. Die beiden Systeme sind sich in ihrer Funktionalität sehr ähnlich.

Zu ihren Merkmalen gehören:

- Speicherung beliebig vieler Versionen einer Datei
- automatische Identifizierung von Versionen
- Verwendung der Deltamethode zur Speicherung der Versionen (die Speicherung von Änderungen einer Basisversion zu einer Änderungsversion, im Gegensatz zur Speicherung der ganzen Datei für jede Version)
- Speicherung von Datum, Grund, versionsabhängigen und versionsunabhängigen Attributen
- Versionsidentifikationen in Sourcedateien, Objektmodulen und gebundenen Programmen
- Koordination von Zugriffen durch mehrere Benutzer

Diese Systeme bieten eine gute Unterstützung für die Verfolgung von Änderungen und zur Aufzeichnung der Historie einzelner Softwarekomponenten. Die Beherrschung von Strukturen und deren Abhängigkeiten in großen Softwaresystemen wird nur marginal unterstützt. Außerdem unterstützen weder *sccs* noch *rcs* das Verwalten von Objektcode oder anderer nicht-ASCII-Dateien.

KMS-X

Um ein leistungsfähiges Konfigurations-Management aufzubauen, müssen die Mechanismen der Versionskontrolle, wie sie von *rcs* und *sccs* angeboten werden, noch mit geeigneten Funktionen erweitert werden. Dieser Weg wurde auch in der Entwicklung des Konfigurations-Management-Systems für SINIX (KMS-X) eingeschlagen.

Die wichtigsten Erweiterungen in KMS-X sind:

- Alle Objekttypen und Dateien — also auch Objektmodule, Binärdateien und Phasen — können mit KMS-X verwaltet werden.
- Automatisches Erstellen und Pflegen von Konfigurationslisten. Mit Hilfe von Konfigurationslisten wird der aktuelle Zustand eines Software-Systems festgehalten. Dadurch wird dieser Stand reproduzierbar gemacht und kann unabhängig von der Weiterentwicklung des aktuellen Systems für sich weiterentwickelt werden.

- Unterstützung strukturierter Hierarchien. Damit können ganze SINIX-Dateiverzeichnissysteme kontrolliert und bearbeitet werden.
- Unterstützung von Varianten. Durch Varianten wird die Entwicklung und Pflege von Softwaresystemen vereinfacht, die aus gemeinsamen und variantenspezifischen Teilen bestehen. Ein Beispiel wäre die Softwareentwicklung für verschiedene Zielsysteme. Bestimmte Softwarekomponenten sind ohne Änderung auf allen Zielsystemen verwendbar (common). Die restlichen Komponenten müssen aber an das jeweilige Zielsystem angepaßt werden. Sie werden dann in KMS-X komfortabel und variantenspezifisch verwaltet.
- Einbindung in vorhandene Umgebungen durch ein Anpassungskonzept. Um mit KMS-X benutzerbezogen arbeiten zu können, wird dem Anwender eine Profil-Ebene zur Verfügung gestellt. Der Anwender kann so sein Profil an seine Arbeitsweise anpassen. Auch der nutzbare Funktionsumfang kann benutzerbezogen konfiguriert werden. In diesem Fall kann der gesamte Funktionsumfang auf den Konfigurationsverwalter oder den Projektleiter beschränkt werden.

Mit KMS-X steht ein flexibles Konfigurations-Management zur Verfügung. Es werden viele Verfahrenselemente angeboten, die an die speziellen Anforderungen und Konventionen eines Projekts angepaßt werden können. Aus diesem Grund kann KMS-X als projektübergreifende Produktbibliothek und auch als lokale Entwicklerbibliothek zum Einsatz kommen. Der Anwender ist in der Lage, KMS-X-Funktionen mit SINIX-Befehlen zu kombinieren. Somit lassen sich die vielfältigen Management-Aktionen für die Konfiguration zu leistungsfähigen Ablaufregeln zusammenfassen, die problemlos in die Softwareentwicklungsumgebung integrierbar sind. KMS-X kann also als zentraler Bestandteil einer Softwareproduktionsumgebung eingesetzt werden.

Zur Zeit wird an der Entwicklung einer OSF/Motif-basierten Bedienoberfläche gearbeitet, die die bisherige Collage-Oberfläche ablösen wird. Weitere Schwerpunkte der Entwicklung sind die Anbindung von KMS-X an OSCE und die Integration mit einem Repository.

CASE-Werkzeuge

Das GRAPES-Designwerkzeug

Es ist bekannt, daß unvollständige bzw. falsche Vorgaben und Fehler in den frühen Designphasen als Hauptfehlerquelle bei der Erstellung von Softwareprodukten anzusehen sind. Um diese Fehlerquellen weitgehend auszuschalten, muß einerseits eine optimale Kommunikation zwischen Anwendern und Entwicklern gewährleistet sein, und andererseits frühzeitig eine rigorose Überprüfung des Designs vorgenommen werden. Dazu bedarf es einer leicht kommunizierbaren aber trotzdem formalen Designmethode. GRAPES ist eine Designmethode, die diese Voraussetzungen erfüllt. Es werden für alle Designaspekte intuitive grafische Repräsentationen angeboten. Gleichzeitig besitzt GRAPES eine formale Syntax und Semantik, auf deren Basis eine optimale Werkzeugunterstützung angeboten werden kann. Der Systemdesigner GRAPES-SD ist die zentrale GRAPES-Komponente mit der Designdokumente erstellt, geprüft und verwaltet werden.

Editorenkomponenten

Um Fehler bereits frühzeitig zu verhindern, besitzt GRAPES-SD Editorenkomponenten, die sich optimal auf jeden Dokumententyp einstellen. Ein Großteil der Fehler wird schon frühzeitig abgefangen, indem das Werkzeug bereits bei der Eingabe nur korrekte Konstrukte zuläßt. Für folgende Diagrammtypen werden u.a. Editorenkomponenten angeboten:

Kommunikationsdiagramme (CD)

Mit der CD-Editorenkomponente können Objekte in Unterobjekte strukturiert und die Kommunikation zwischen den Unterobjekten festgelegt werden. Der Systemdesigner kann kompatible Kommunikationsschnittstellen für die Unterobjekte erzeugen.

Prozeßdiagramme (PD)

Prozeßabläufe werden wohlstrukturiert entworfen. Vorgefertigte Ablaufkonstrukte können ausgewählt und in das Prozeßdiagramm eingefügt werden. Das grafische Layout wird automatisch neu berechnet.

Spezifikationsdiagramme (SD)

Durch Anlegen von Prozeduren, Parametern, Variablen und Konstanten wird die Exportschnittstelle eines Moduls beschrieben.

Entity-Relationship-Diagramme (ER)

Es wird eine Datenmodellierung nach der ER-Methode unterstützt. Entities und Relationships können frei plaziert werden. Die Angaben zu Relationships, wie z.B. Kardinalitäten, werden syntaktisch überprüft.

Alle Editorenkomponenten des GRAPES-SD sind mit einer komfortablen grafischen Bedienoberfläche ausgestattet (GUI), die die Vorteile der Mehrfenstertechnik voll zur Geltung bringt. Dadurch können Designdokumente in parallelen Fenstern betrachtet bzw. bearbeitet werden (z.B. ein Dokument, in dem eine Variable vorkommt, parallel zum Dokument, das die Variable definiert). Die Copy/Paste-Funktion ermöglicht ein fensterübergreifendes Kopieren. Das Design erfolgt auf einer „unendlichen" Zeichenfläche und kann stufenlos vergrößert oder verkleinert werden. Symbole und Linien sind direkt mit der Maus manipulierbar. Der Anwender wird durch zahlreiche Generierungsleistungen entlastet. Sollen zum Beispiel zwei Symbole durch eine Linie verbunden werden, reicht es aus, nur die betroffenen Symbole mit der Maus anzuklicken; das Werkzeug berechnet automatisch den Verlauf der Verbindung, ohne daß dabei Symbole geschnitten werden. Beim Verschieben von Symbolen werden alle Verbindungen vom und zum Symbol mit verschoben. Die Beschriftung von Verbindungen wird automatisch vom Werkzeug plaziert. Der Modellierer kann das Layout jederzeit automatisch berechnen lassen. Bei Änderungen im Text werden die grafischen Symbole an die Textmenge angepaßt. Gelangt man beim Bearbeiten des Designs an die Grenzen des sichtbaren Bereichs, wird der Bereich automatisch verschoben. Für die Texte stehen hilfreiche Such- und Ersetzungsfunktionen zur Verfügung. Außerdem gibt es eine mehrstufige Undo-Funktion. Häufig benötigte Funktionen werden als Buttons in einer „Toolbox" angeboten. Die Help-Funktion unterstützt den unerfahrenen Anwender.

Der Modelleditor

Trotz dezidierter Editorenkomponenten besitzt GRAPES-SD eine über alle Dokumenttypen integrierte Datenhaltung. Die Integration der Einzeldokumente zu einem Gesamtmodell erfolgt mit dem Modelleditor. Mit ihm werden Modelle aus Designdokumenten hierarchisch aufgebaut.

- Eine wesentliche Komponente des Modelleditors ist der Editor für Hierarchiediagramme (HD), mit dem die Dokumentenhierarchie erstellt und gepflegt wird. Das Hierarchiediagramm ermöglicht einen schnellen Zugriff auf die Designdokumente. Mit dem HD-Editor können Umbenennungen über das gesamte Design hinweg vorgenommen werden.
- Ein zweiter Aspekt des Modelleditors ist die Arbeitsstandsicherung. Startet ein Anwender GRAPES-SD erneut, so gelangt er genau in den Arbeitsstand, mit dem er den Editor zuletzt verlassen hatte. Die Arbeitsstandsicherung umfaßt auch einen benutzerbezogenen Arbeitsbereich für die in Arbeit befindlichen Dokumente (Scratchdiagramme) und Referenzen in das Modell. Die Dokumente und Referenzen werden im Arbeitsbereich als Icons dargestellt, die mit Mausklick geöffnet und dann bearbeitet werden können.
- Der dritte Aspekt ist die Mehrbenutzerfähigkeit des Modelleditors. Während der Bearbeitung werden Teilmodelle kurzfristig gesperrt. Wenn zu dieser Zeit ein konkurrierender Benutzer auf das Teilmodell zugreifen will, wird der Zugriff verweigert. Dadurch wird verhindert, daß die Änderungen eines Designers durch konkurrierende Bearbeitung verloren gehen.

Jedem GRAPES-Designdokument können kundenspezifische Dokumente assoziiert werden. Wenn solche Dokumente ausgewählt werden, werden die konfigurierten externen Werkzeuge angeboten. Bei Auswahl des externen Werkzeugs wird die Bearbeitung des kundenspezifischen Dokuments gestartet. Die Integration und Verwaltung der geänderten Dokumente erfolgt innerhalb der GRAPES-Datenhaltung.

Modellanalyse

Neben der dokumentspezifischen Prüfung zum Eingabezeitpunkt wird im GRAPES-SD eine systemweite Modellanalyse angeboten, die vom Anwender explizit gestartet werden kann. Die Prüfkriterien der Modellanalyse sind Konsistenz, Vollständigkeit und Korrektheit. Viele Designfehler können durch diese statische Methode entdeckt werden. Beispiele dafür sind:

- Werden Prozeduren, Variablen oder Konstanten eines Moduls korrekt verwendet?
- Findet zwischen zwei Prozessen ein korrekter Nachrichtenaustausch statt?
- Werden Prozeduren, Parameter, lokale Variablen, usw. jemals benutzt?

In einer Fehlerliste werden die erkannten Fehler und Warnungen festgehalten und können mit einem Fehlerbrowser abgearbeitet werden. Dabei werden die Fehler nacheinander präsentiert und zu jedem Fehler die betreffenden Diagramme bzw. Diagrammpaare ausgegeben.

Auskunftsfunktion

Während der Arbeit mit dem Systemdesigner steht jederzeit eine integrierte Auskunftsfunktion zur Verfügung. Sie informiert, wo die angegebenen Designelemente verwendet werden (Querverweise). Außerdem werden Designstrukturen in übersichtlicher und konzentrierter Form angezeigt. Die Ergebnisse der Auskunftsfunktion können als Ausgangspunkt weiterer Untersuchungen dienen.

Druckfunktion

Die Druckfunktion von GRAPES-SD führt eine automatische Paginierung der Designdokumente durch. Neben dem Posterdruck, bei dem das Dokument unverändert in Druckseiten aufgeteilt wird, steht ein Konnektorendruck zur Verfügung. Bei dieser Art der Druckaufbereitung werden Konnektorsymbole erzeugt, wenn Verbindungen über eine Druckseite hinausgehen und die Druckseite wird mit einem Rahmen versehen. Solche Konnektordrucke eignen sich besonders für das Abheften in technischen Dokumentationen. Vor dem Ausdrucken können die Ausgaben mit einer Preview-Funktion kontrolliert werden. Als Ausgabeformate stehen Postscript und CGM zur Verfügung.

Verbindungen zu Textsystemen

Für die Textsysteme HIT und FrameMaster werden Kopplungsbausteine angeboten. GRAPES-Dokumente können in Textdokumente dieser Systeme integriert werden (Mischdokument). Der Kopplungsbaustein garantiert, daß bei Änderungen der GRAPES-Dokumente die Mischdokumente aktualisiert werden.

Datenmodellierung

Mit GRAPES kann ein Datendesign mit den Mitteln der ER-Modellierung erstellt werden. Die Weiterverarbeitung des Datendesigns erfolgt mit dem Werkzeug RDDG. Dazu wird das ER-Modell geprüft und normalisiert. Sind Änderungen des ER-Modells nötig, werden die Änderungen interaktiv mit den Mitteln des GRAPES-SD durchgeführt. Normalisierte ER-Modelle können in SQL-Datenbankbeschreibungen umgesetzt werden.

Modellinterpreter

Es besteht die Möglichkeit, GRAPES-Modelle mit Hilfe eines Interpreters „auszuführen". Unter Verwendung geeigneter Modellierungstechniken (Z-Modellierung) liegen ausführbare Modelle bereits in ganz frühen Stadien des Designs vor. Die Kommunikation zwischen den Anwendungsexperten und den Entwicklern wird dadurch wesentlich effektiver. Zudem liefert die Modellinterpretation die erforderlichen Informationen, um semantische Fehler im Design frühzeitig erkennen zu können. Bei der Modellinterpretation wird eine Unterscheidung zwischen Simulation und Ausführung eines Modells getroffen.

Das Ziel der Simulation ist das Erkennen von Schwachstellen. Deshalb wird ein Modell ausgeführt und die Ergebnisse werden dann unter speziellen Gesichtspunkten, wie z.B. Häufigkeiten und Ablaufzeiten, betrachtet.

Ziel der Modellausführung ist es, semantische Fehler des Designs zu erkennen und zu lokalisieren. Während einer Modellausführung werden der Ablauf sowie die Zustände und Datenbelegungen während des Ablaufs angezeigt. Die Modellausführung kann auch gezielt angehalten werden.

Generierung

Ein mit GRAPES entworfenes Design kann in Code umgewandelt werden. Zur Zeit wird daran gearbeitet, eine Umsetzung anzubieten, die speziell auf OLTP-Anwendungen ausgerichtet ist.

OSCE: Eine integrierte Software-Entwicklungsumgebung für SINIX

Das ECMA-Referenzmodell für CASE, ein Modell für eine neue Generation von Software-Entwicklungsumgebungen, hebt besonders die Bedienoberfläche und die Integrationskontrolle für CASE-Werkzeuge sowie die Datenintegration hervor. Mit dem OSCE-System (**O**pen **S**ystems **C**ASE **E**nvironment) bietet Siemens Nixdorf eine offene, verteilte, OSF/Motif-basierte Entwicklungsumgebung für SINIX an, die sich auf das ECMA-Konzept stützt.

Ein Blick auf die heutige Spitzentechnologie bei den CASE-Werkzeugen in der Computerindustrie zeigt, daß die Einschränkungen bei der Integration von Werkzeugen den breiteren Einsatz von CASE-Werkzeugen verhindern. Viele CASE-Werkzeuge sind fest in geschlossene Umgebungen integriert. Sie verfolgen ganz spezifische Ansätze und schließen weitgehend die Verwendung anderer, insbesondere kundenspezifischer Werkzeuge in der gleichen Umgebung aus. Eine flexible Konfigurierung der Software-Entwicklungsumgebung zur Anpassung an die unterschiedlichsten Kundenbedürfnisse ist nur auf der Basis eines offenen Systems wie OSCE möglich.

OSCE ist eine offene Software-Entwicklungsumgebung für 3GL-, 4GL- und 00-Programmiersprachen, die auf internationalen Standards (UNIX, NFS, OSF/Motif) basiert. Die Architektur von OSCE realisiert bereits in Teilen das ECMA-Referenzmodell für offene CASE-Umgebungen. Dieses Referenzmodell, das die Basis für die Entwicklung weiterer Standards auf dem CASE-Sektor sein soll, beschreibt verschiedene Aspekte: Integration von Bedienoberflächen, Inter-Tool-Kommunikation über Messages (beides in OSCE realisiert) sowie die Daten- und Prozeßintegration.

OSCE unterstützt die effiziente Entwicklung komplexer Anwendungen in einer verteilten Entwicklungsumgebung und erlaubt die einfache Integration kundenspezifischer Werkzeuge. OSCE ermöglicht Interworking mit den Produkten SoftBench (Hewlett-Packard), OpenCase/ToolBus (INFORMIX), SDE (IBM) und anderen in

heterogenen Netzen. Insgesamt bietet OSCE alle wesentlichen Voraussetzungen für die Steigerung der Produktivität im Software-Entwicklungsprozeß. OSCE besteht aus den Komponenten OSCE-ToolBus und dem optionalen OSCE-Encapsulator.

OSCE-ToolBus

Allgemeine Eigenschaften

In der Basiskonfiguration bietet OSCE die komfortable Unterstützung des Editier-Compilier-Zyklus für C, C++, FORTRAN und Pascal, außerdem Funktionen des Konfigurations-Managements und für die Kommunikation im Team. Die wesentlichen Eigenschaften des Systems werden in den folgenden Abschnitten beschrieben.

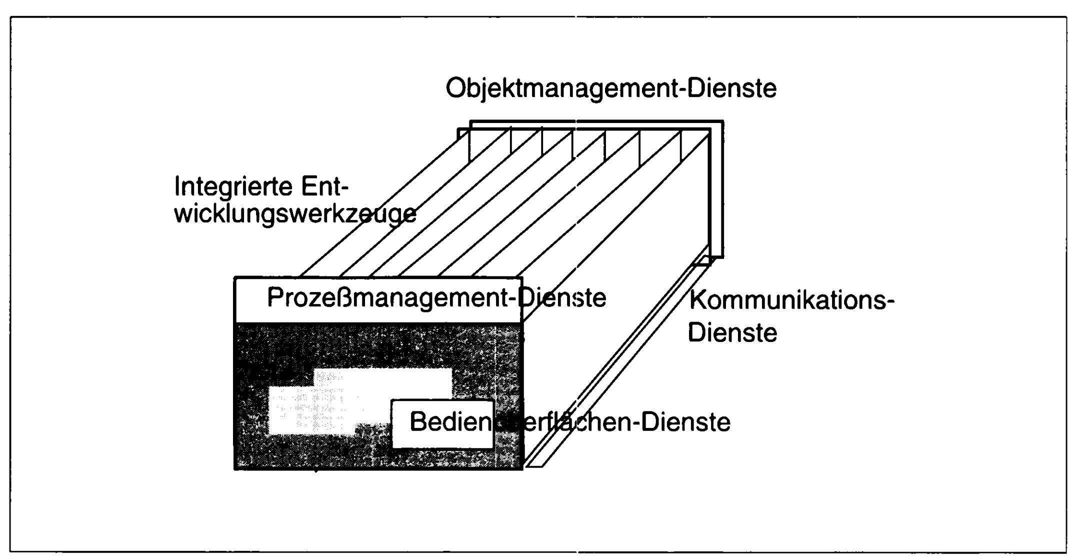


Abbildung 5-3: Das in OSCE (SINIX) realisierte ECMA CASE-Referenzmodell.

Kommunikation der Werkzeuge

Die Werkzeuge kommunizieren über einen **B**roadcast **M**essage **S**erver (BMS) miteinander. Das erlaubt eine weitgehende Automatisierung von Verfahrensabläufen und damit ein aufgabenorientiertes Arbeiten. Für den Anwender steht nicht mehr so stark im Vordergrund, welche Kombination von Werkzeugen er zur Erledigung einer bestimmten Aufgabe braucht. Vielmehr kann er sich voll auf seine eigentlichen Aufgaben im Entwicklungsprozeß konzentrieren.

Verteilte Verarbeitung

OSCE unterstützt die Verarbeitung aller Daten im UNIX-Netz. Zur Bearbeitung der Daten können beliebige OSCE-Werkzeuge auf verschiedenen Rechnern im Netz genutzt werden. Die Ausgabe eines Werkzeugs kann auf jedes Ausgabegerät im Netz umgelenkt werden. Die Arbeiten mehrerer Entwickler an einem Projekt können mit den integrierten Funktionen des Konfigurations-Managements koordiniert werden. Es können auch Server-Konzepte, wie z.B. File-Server oder Compile-Server, realisiert werden.

Bedienoberfläche

Die in OSCE integrierten Werkzeuge verfügen über eine einheitliche und leicht bedienbare grafische Benutzerschnittstelle auf der Basis von OSF/Motif. Ein spezieller Help Manager steht zur komfortablen Ausgabe von Online-Hilfsinformationen zu den OSCE-Werkzeugen zur Verfügung. Die Menüs dieser Werkzeuge können durch den Benutzer erweitert werden, so daß eigene Werkzeuge aus den vorhandenen heraus aufgerufen werden können.

Integrierte Werkzeuge

Bereits in der Basiskonfiguration enthält OSCE eine Reihe leistungsfähiger Werkzeuge, die im folgenden beschrieben werden.

Tool Manager

Der Tool Manager erlaubt das Starten und Beenden von Werkzeugen und gibt Auskunft über den Status der momentan aktiven Werkzeuge. Außerdem erlaubt er das Abspeichern und Wiederherstellen von Werkzeugkonfigurationen.

Development Manager

Mit dem Development Manager kann man sich einen Überblick über Dateisysteme (inklusive verteilter Dateisysteme) verschaffen. Außerdem unterstützt er die objektorientierte Verarbeitung der gespeicherten Daten. Er verfügt über einen Anschluß an das integrierte Konfigurations-Managementsystem, das entweder *sccs*- oder *rcs*-basiert sein kann. Von diesem Punkt aus können auch weitere wichtige Funktionen anderer OSCE-Werkzeuge, wie z.B. Static Analyzer oder Program Builder, aufgerufen werden.

Program Builder

Der Program Builder erlaubt das automatische Generieren von Makefiles sowie deren anschließende Ausführung. Bei der Ausführung von Makefiles werden Fehlermeldungen in einem eigenen Fenster angezeigt. Nach Selektion einer Fehlermeldung (mit Mausklick) wird automatisch der Editor aktiviert und zeigt die fehlerhafte Zeile im betroffenen Programmcode an, so daß der Fehler unmittelbar korrigiert werden kann.

Program Editor

OSCE hat einen eigenen grafischen, syntaxsensitiven Editor. Wenn eine Datei in OSCE geändert wird, wird der geänderte Dateiinhalt automatisch in allen Fenstern, die auch die Datei anzeigen, dargestellt. Der Editor in OSCE kann konfiguriert werden. So können beispielsweise die Editoren *MAXed*, *vi* oder *emacs* eingesetzt werden.

Static Analyzer

Mit Hilfe des Static Analyzer kann in OSCE nach Definitionen, Deklarationen von Funktionen und Variablen sowie nach Referenzen von Funktionen und Variablen gesucht werden. Der betreffende Programmcode wird dann automatisch im Editor angezeigt. Die Funktionen des Static Analyzers können auch aus dem Editor heraus aufgerufen werden. Insbesondere bedeutet das, daß der Static Analyzer eine sehr effiziente Unterstützung der Analyse von Programmstrukturen vorhandener Programme anderer Entwickler anbietet.

Mail

Die Einbettung von *mailx* als OSCE-Werkzeug erlaubt dem Entwickler einen komfortablen Zugang über die grafische Bedienoberfläche und Menüs zu Mail-Funktionen und erleichtert so die Kommunikation im Team.

OSCE-Encapsulator

OSCE stellt auch eine Integrationsplattform für die Integration von Standard-UNIX-Werkzeugen oder kundenspezifischen Werkzeugen in die Entwicklungsumgebung dar. Das Integrationswerkzeug dafür ist der OSCE-Encapsulator. Mit dessen Hilfe hat der Anwender die Möglichkeit, z.B. Standard-UNIX-Programme (*find*, *grep*, usw.) oder eigene Werkzeuge zu integrieren. Kommando- und *terminfo*-basierte Anwendungen erhalten so eine moderne Fensteroberfläche, ohne daß Änderungen im Sourcecode notwendig werden. Über eine offengelegte Programmschnittstelle (C/C++) können auch vorhandene OSF/Motif-Anwendungen integriert werden. Durch Nutzung des vorhandenen Triggermechanismus auf der Basis von Events kann der Software-Entwicklungsprozeß weiter rationalisiert werden, z.B. durch Automatisierung immer wiederkehrender Abläufe.

OSCE-Erweiterungen

Mit dem Produkt OSCE-Link besteht die Möglichkeit, den Komfort und die hohe Funktionalität von OSCE sowie von anderen in OSCE integrierten Werkzeugen auch für Entwicklungen im BS2000 zu nutzen. Daneben bietet Siemens Nixdorf eine Reihe von integrierten Werkzeugen für OSCE an. Hier sind insbesondere die Produkte I4GL for ToolBus zur Unterstützung der INFORMIX-4GL-Anwendungen und der C++ Developer als grafischer Klassenbrowser für C++ zu nennen. Weitere Beispiele sind KMS-X, DBX und MAXed. Die Liste dieser Werkzeuge wird ständig erweitert.

6 Systemverwaltung und Netzmanagement

Der Einsatz von Rechnern und Netzwerken in entscheidenden Funktionen in Unternehmen, Behörden, öffentlichen und sonstigen Einrichtungen wächst zusehends. Das bedeutet, daß Systemfehler und Fehler in Netzwerken zu nicht vertretbaren Verzögerungen bei Lieferungen und im Service führen, Produktionseinbußen und Wettbewerbsnachteile nach sich ziehen und damit auch zu finanziellen Verlusten des Unternehmens führen. Diese Situation stellt hohe Anforderungen an die Verfügbarkeit solcher Systeme. Fehlerdiagnose und Störungsbeseitigung werden jedoch durch Größe und Komplexität der Systeme und Netze sowie durch die örtliche Verteilung der einzelnen Komponenten erheblich erschwert.

Geringe Durchsatzraten und lange Wartezeiten für Anwender sind häufig auf Engpässe bei den Ressourcen zurückzuführen. Diese Engpässe müssen rechtzeitig erkannt und behoben werden, zum Beispiel durch eine Neukonfigurierung des Netzes.

Die Aufgaben der Systemverwaltung und des Netzmanagements reichen von der Planung und Konfiguration über Lastmessung, Erstellen von Statistiken, Fehlerdiagnose und Fehlerbeseitigung bis hin zur Software-Freigabe, ihrer Verteilung und Verwaltung. Systemverwaltung und Netzmanagement umfassen die Mechanismen, Werkzeuge und Produkte, die erforderlich sind, um Systeme und Netze zu steuern und zu überwachen.

Systemverwaltung

Die Aufgaben eines Systemverwalters erstrecken sich auf alle Phasen der Laufzeit eines Rechners. Sie reichen von der Installation des Systems über die Wartung während der Laufzeit bis zum Hinzufügen von Komponenten zum existierenden System. Typische Aktivitäten in den einzelnen Phasen sind:

- Installation
  Die erste Aufgabe besteht darin, die Hardware und Software des Systems zu konfigurieren. Das heißt, die Basis-Software einzubringen und das System an existierende Wide-Area- oder Local-Area-Netzwerke anzuschließen.
- Wartung während des Betriebs
  Dies umfaßt präventive Wartungsmaßnahmen wie das Überwachen des Dateisystems oder das Anlegen von Sicherungen. Hinzu kommt eine Reihe von Aufgaben wie das Eintragen von neuen Benutzern, das Installieren von neuer Anwender-Software und vieles andere mehr.
- Installieren von neuen Hardware-Komponenten
  Wenn neue Hardware erworben wurde, liegt es in der Verantwortung des Systemverwalters, diese Geräte anzuschließen und zu konfigurieren, ohne den laufenden Betrieb signifikant zu beeinträchtigen.

OA&M

Die OA&M-Schnittstelle (**O**peration, **A**dministration and **M**aintenance) des SINIX-Systems ist ein Zugang zu den wesentlichen Verwaltungsfunktionen des Systems. OA&M ist eine zeichenbasierte, menü- und maskenorientierte Oberfläche, die leicht zu erlernen und anzuwenden ist. Zur Zeit werden mit der OA&M-Software eine deutsche und eine englische Version ausgeliefert. Andere Sprachvarianten lassen sich ohne Änderung der Bedienoberfläche erzeugen.

Unter anderem bietet OA&M folgende Funktionen an:

- Benutzerverwaltung
- Verwaltung von Dateisystemen
- Einstellen von Systemparametern (z.B. Uhrzeit)
- Performance-Analysen, wie etwa den CPU-Zeitverbrauch des Systems
- Anwenderschnittstelle zur Konfigurierung der Hardware

Zusätzlich zu diesen Standardfunktionen bietet OA&M eine offene Schnittstelle an, über die sich weitere Pakete in die Anwenderschnittstelle integrieren lassen, wie z.B. die Verwaltung des Sicherungssystems Backup oder des Siemens Nixdorf SPOOLs (siehe Seite 104 in diesem Kapitel und im Abschnitt „SPOOL: Drucken in Systemen mit verteilter Verarbeitung" auf Seite 130).

Config

Config ist ein Konfigurationssystem zur komfortablen Hardware-Konfigurierung. Unter Konfiguration von Hardware versteht man die Gesamtheit der Verwaltungstätigkeiten, die notwendig sind, um periphere Geräte an einen Rechner anzuschliessen und zu betreiben. Das Anschließen von Peripheriegeräten ist eine der komplexesten Aufgaben der Systemverwaltung. Standardisierte Abläufe und Verfahrensweisen sind dabei nicht möglich, weil es eine Vielzahl unterschiedlicher Geräte gibt, sowie eine große Zahl von Anschlußmöglichkeiten und verwendbaren Protokollen. Die Hauptaufgabe bei der Konfiguration von Hardware besteht darin, die unterschiedlichen Systemparameter konsistent zu halten. Geräte müssen eingestellt (z.B. landesspezifische Tastaturtabellen laden) oder am Rechner eine Reihe von Anpassungen durchgeführt werden (z.B. die Pflege der Systemdatei */etc/hosts* beim Anschluß von X-Terminals). So sind folgende Schritte (neben anderen) auszuführen, um einen Drucker an einen Rechner anzuschließen:

- Eintragen des Drucker in die Konfigurationsdatei des Spoolers (Druckertyp, verwendete Emulation, verwendete Code-Tabellen, usw.)
- Anlegen eines */dev*-Knotens mit einer bestimmten Major- und Minor-Gerätenummer. Diese sind abhängig von der verwendeten Systemschnittstelle und der verwendeten Kanalnummer.
- Einstellen der *tty*-Parameter für den Knoten. Diese sind abhängig von der verwendeten Systemschnittstelle und dem angeschlossenen Druckertyp.
- Zuordnen des */dev*-Knotens für den jeweiligen Drucker in Beschreibungsdateien des Spool-Systems.

Die einzustellenden Parameter sind nicht immer die gleichen. Sie sind abhängig vom Drucker, von der benutzen Anschluß-Schnittstelle, von der Software, die zum Betrieb des Druckers eingesetzt wird, und anderem mehr.

Das Beispiel zeigt, daß das Einstellen gültiger Geräte- und Systemvariablen bestimmten Bedingungen entsprechen muß, wenn ein fehlerfreies Verhalten des Systems gewährleistet sein soll. Diese Bedingungen legen für korrekte Systemparameter Wertebereiche fest und bestimmen die Abhängigkeiten zwischen den einzelnen Systemparametern. Die Zusammenhänge sind oft so komplex, daß nur Systemspezialisten in der Lage sind, ohne Software-Unterstützung eine fehlerfreie Konfiguration von Hardware-Komponenten vorzunehmen.

Das Ziel von Config ist es, die für das Anschließen und Konfigurieren von peripheren Geräten notwendigen Aufgaben so weit wie möglich zu automatisieren, um so Parameter fehlerfrei und konsistent einzustellen. Config erlaubt es, sowohl die Wertebereiche und Abhängigkeiten von Systemparametern als auch die Struktur der angeschlossenen Peripherie extern zu beschreiben. Damit ist es möglich, neue Geräte in das Konfigurationssystem einzuführen und die Eigenschaften bestehender Geräte zu verändern.

Architektur und Schnittstellen

Bei dem Design des Konfigurationssystems Config wurden folgende Anforderungen berücksichtigt:

- Das Konfigurierungswerkzeug soll erweiterbar und anpassungsfähig sein. Das Konfigurierungswerkzeug muß sich leicht an die verschiedenen peripheren Geräte und an neue Hardware anpassen lassen. Sowohl die Definition von Konfigurationsabläufen entsprechend der bestehenden Hardware als auch die Anpassung des Werkzeugs selbst an neue Zentraleinheiten und neue Betriebssystemversionen muß leicht und ohne großen Aufwand durchführbar sein. Aus diesem Grund sind die Konfigurationsabläufe von der aktuellen System-Hardware zu trennen und in einer speziellen Regelsprache zu implementieren.
- Das Konfigurierungswerkzeug soll grafische und alphanumerische Bedienoberflächen unterstützen.
- Das Konfigurierungswerkzeug soll es ermöglichen, Konfigurationen für Fremdrechner auf einem zentralen Konfigurationsrechner vorzubereiten. Derartige Konfigurationsmodelle können über ein Service-Netz zu den Kundenrechnern transferiert werden oder vor Ort von einem Service-Techniker manuell eingelesen werden. Das bedeutet für die Architektur des Systems, daß die Konfigurationsdaten nicht unmittelbar in das System übernommen werden dürfen. Die Spezifikation der Konfiguration und die Übernahme der Konfigurationsdaten in das System sollen voneinander getrennt werden.

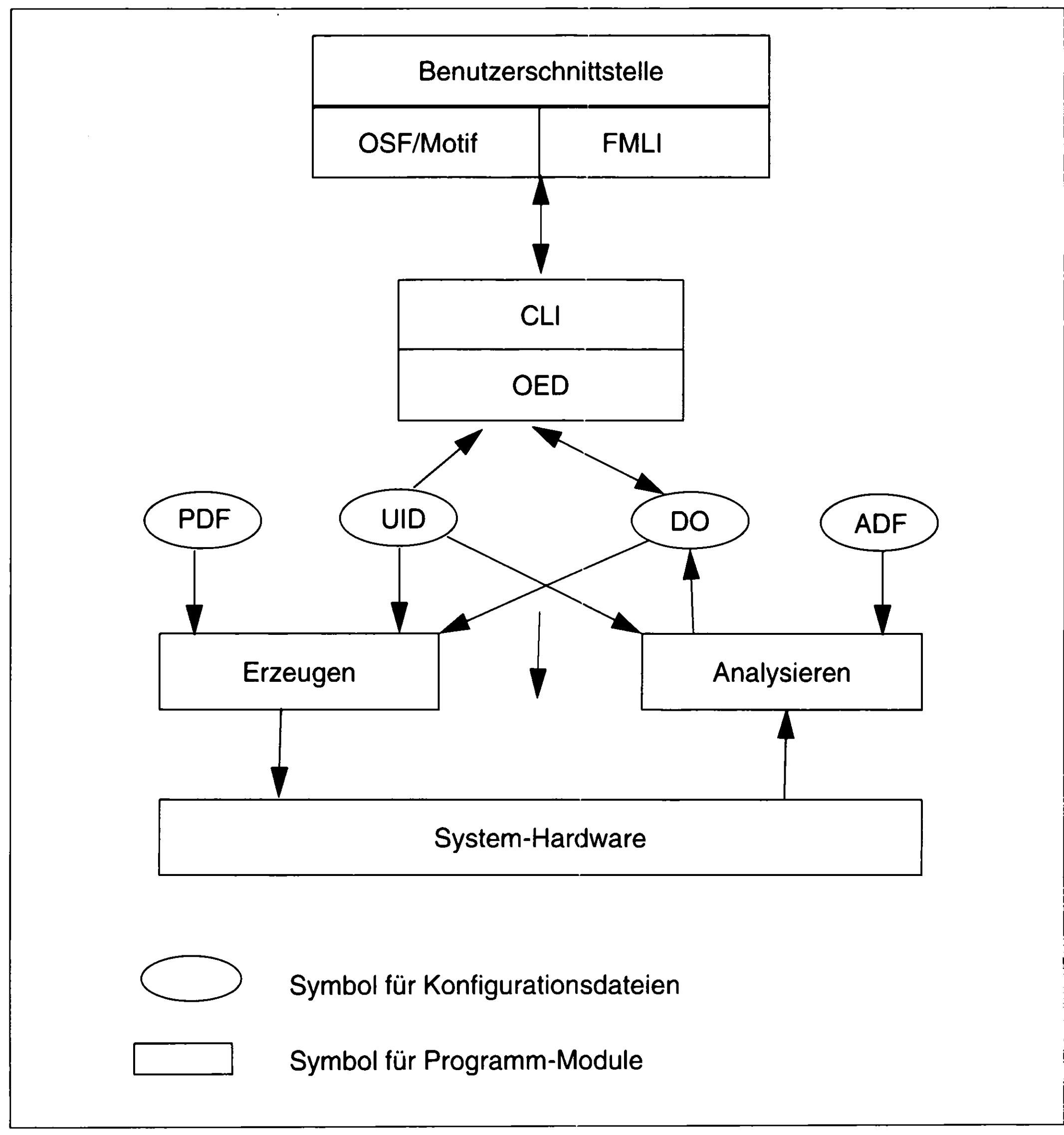


Abbildung 6-1: Die Systemarchitektur von Config (SINIX).

Diese Anforderungen führen zu einer Systemarchitektur, wie sie in Abbildung 6-1 dargestellt ist.

Die Benutzerschnittstelle zeigt dem Benutzer die Konfigurationsdaten. Diese sind objektorientiert angelegt. Die Objekte sind in einem Objektbaum hierarchisch angeordnet, der den logischen Zusammenhang der zu konfigurierenden Objekte widerspiegelt, und sowohl auf der alphanumerischen Oberfläche als auch innerhalb der grafischen Oberfläche angeboten wird (Siehe Abbildung 6-2).

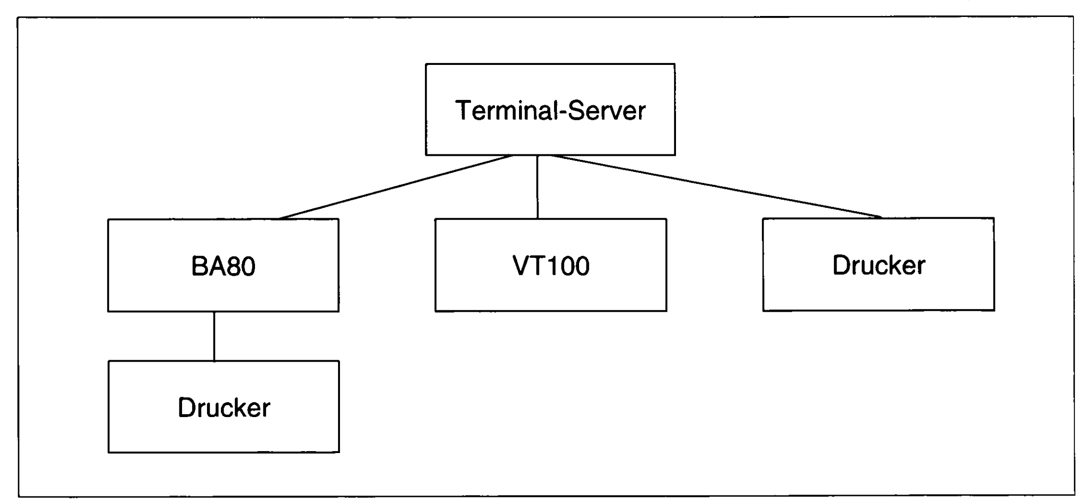


Abbildung 6-2: Beispiel eines Config-Objektbaums

Der Anwender kann innerhalb des Baums mit verschiedenen Methoden navigieren. Er kann die angezeigten Konfigurationsobjekte und deren Eigenschaften mit den Operationen *add, remove* und *open* verändern.

Die zentral steuernde Komponente des Systems ist der Objekt-Editor (**O**bject **Edi**tor - OED). Über die Kommandoschnittstelle (**C**ommand **L**ine **I**nterface - CLI) empfängt er von der Bedienoberfläche Befehle zur Manipulation des Objektbaums (**D**ata **O**bject - DO). Kontrollinformationen und Ausgabedaten gibt der Objekt-Editor an die Bedienoberfläche zurück. Operationen am Objektbaum dürfen nur dann durchgeführt werden, wenn die Konsistenz des Datenbestandes gesichert ist. Der Objekt-Editor verwendet zwei externe Datenstrukturen, um die Konsistenz zu prüfen:

Hardware-Beschreibungsdatei (UID)

In dieser UID-Datei (**U**ser **I**nterface **D**efinition) ist die Struktur der Hardware beschrieben, die konfiguriert werden soll. Es können die Abhängigkeiten von Werten beschrieben werden und es stehen Sprachmittel zur Verfügung, mit denen die Konsistenz der Daten geprüft werden kann. Die Funktionalität der Beschreibungssprache erlaubt es, die Konfigurationsobjekte auf hierarchische Weise zu beschreiben.

Datenobjekte (DO)

Die DO-Dateien sind eine semantische Kopie der aktuellen Systemparameter und beschreiben vollständig die Hardwarekonfiguration des Systems. Die aktuellen Systemwerte sind entsprechend der UID-Beschreibung strukturiert. Man kann sich die Datenobjekte als eine Instanziierung der UID-Dateien vorstellen. Das Datenobjekt wird entweder manuell durch den Benutzer erzeugt oder automatisch durch das Ausführen eines Analysierungslaufs. Die Bearbeitung der Systemparameter durch den Objekt-Editor verändert die Werte in den Datenobjekt-Dateien. Nach Beenden der Konfigurationssitzung (oder während der Sitzung durch Ausführen eines entsprechenden Kommandos) werden die in den Datenobjekten vorhanden Werte in das System übernommen.

Mit diesen beiden Strukturen teilt der Objekt-Editor der Benutzerschnittstelle während der Laufzeit alle notwendigen Präsentationsinformationen mit. Die Informationen umfassen:

- Attribute des aktuellen Objekts sowie gültige Wertebereiche pro Attribut. Die Entscheidung, welches Präsentationsmittel zur Darstellung eines Attributwertes verwendet wird, wird autonom von dem Benutzerschnittstellen-Modul getroffen.
- Fehlermeldungstexte und Hilfetexte. Durch die Verwendung von NLS (XPG3) lassen sich verschiedene Sprachvarianten erzeugen.
- Der Objekt-Editor stellt bei der Eingabe neuer Attributwerte über die Bedienoberfläche die Korrektheit der neuen Werte her, und zwar mit Hilfe der aktuellen Werte in den DO-Dateien und der in der UID formulierten Konsistenzprüfungen. Fehlerhafte Werte werden zurückgewiesen oder in das Datenobjekt übernommen, anschließend wird das gesamte Datenobjekt als für einen Erzeugungslauf nicht geeignet gekennzeichnet.

Backup

SINIX-Systeme haben exzellente Netzwerkeigenschaften und sind deshalb oft in Netzwerken installiert. Das macht den Einsatz von verteilten Ressourcen in einem Umfang möglich, der vorher nicht möglich war. In diesem Zusammenhang ist der Einsatz von verteilten Datensicherungsdiensten eine unabdingbare Voraussetzung für die heutige Systemverwaltung.

Die Verwendung von verteilten Ressourcen meint nicht einfach den Zugriff auf entfernte Daten und Geräte. Um die notwendige professionelle Datensicherung in einem Netzwerk durchzuführen, ist die Integration von Diensten wie netzwerkweite Auftragsverwaltung, Geräteverwaltung, Medienverwaltung und Benutzerverwaltung genauso wichtig wie leichte, flexible Konfiguration und ein hoher Grad an Verfügbarkeit. In einer kommerziellen Umgebung sind Kriterien wie einfache Bedienung, effektive Meldungsmechanismen, hoher Datendurchsatz, Kostenminimierung und Sicherungsläufe ohne Bediener-Eingriff schon immer entscheidend. Dieses Szenario beschreibt die Anforderungen an Backup, das netzwerkweite Datensicherungswerkzeug für SINIX-Systeme.

Die Architektur

Die Architektur von Backup basiert auf einem Client-Server-Modell, das ein Maximum an Flexibilität ermöglicht. Die einzelnen Komponenten des Werkzeugs können auf separaten Systemen laufen, die in ein Netzwerk eingebunden sind. Der Unterschied zwischen lokalen und entfernten Operationen ist für den Client aufgrund der einheitlichen Syntax nicht sichtbar.

Das Herz von Backup ist eine verteilte Datenbank. Diese Datenbank enthält alle für die Datensicherung relevanten Informationen über das Netzwerk in Form von Objekten. Nur ein System im Netz hat zu jeder Zeit schreibenden Zugriff auf die Datenbank. Dieses System ist der sogenannte „Backup Master“. Alle Änderungen in der Datenbank werden vom Master-Rechner ausgeführt. Auf diese Weise ist die Konsistenz der Daten in der Datenbank gewährleistet. Damit allen beteiligten Systemen im Netz ein schneller Lesezugriff auf die Objekte garantiert werden kann, erhält jedes System eine Kopie der Datenbank. Fällt der Rechner mit der Master-Datenbasis aus, übernimmt ein beteiligter Rechner die Rolle des Masters. Dies geschieht auf der Basis einer festgelegten Priorität.

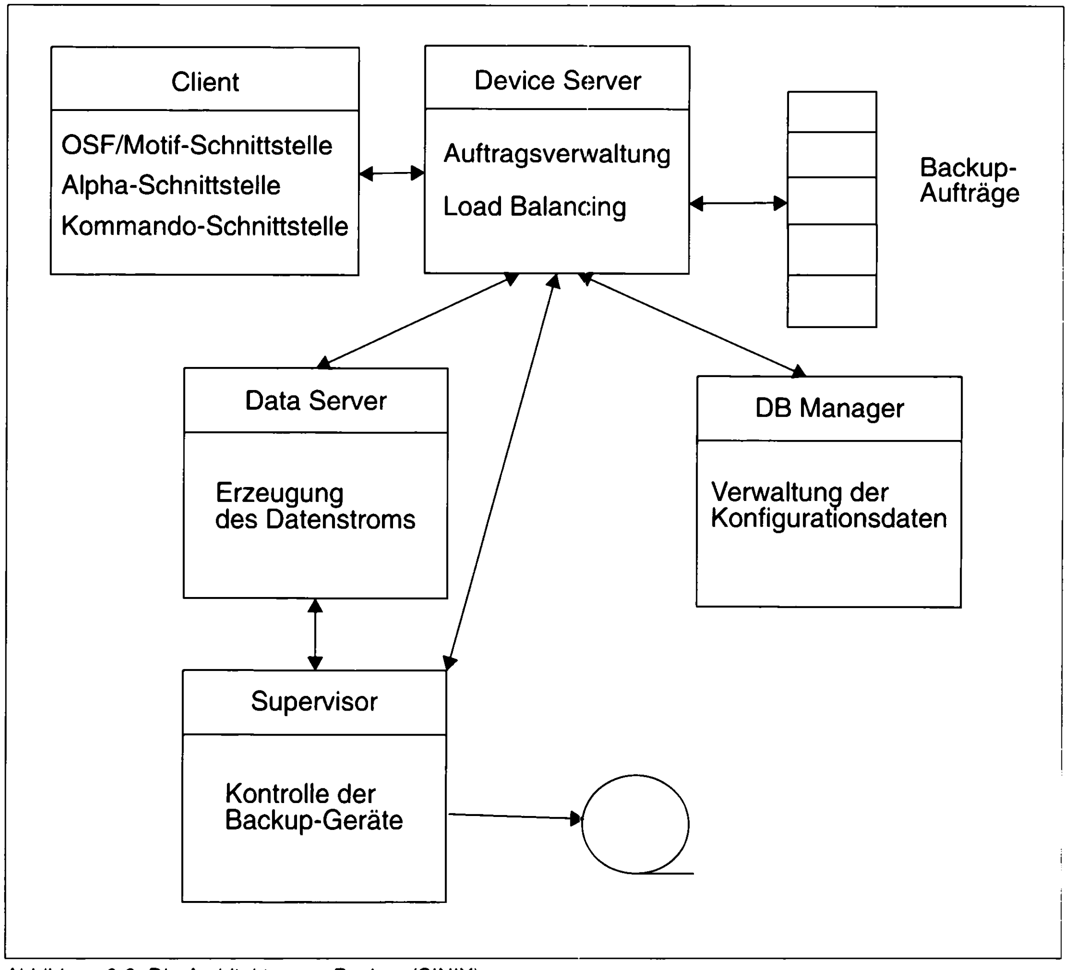


Abbildung 6-3: Die Architektur von Backup (SINIX).

Gleiche Sicherungsgeräte sind in Gruppen zusammengefaßt und werden von einem Geräte-Server verwaltet. Der Geräte-Server erlaubt parallele Sicherungsaufträge, und die Lastverteilung erfolgt auf Sicherungsgeräten der Gruppe. Der Geräte-Server verwaltet eine Warteschlange der Sicherungsaufträge mit einer Supervisor-Nummer. Er koordiniert die Sicherungsaufträge und stellt die Verbindung zwischen dem Daten-Server und dem Supervisor her. Die Lastverteilung wird durch die Aufteilung der laufenden Sicherungsaufträge auf den Supervisor erreicht. Alle Informationen, die Datenserver und Supervisor brauchen, werden von der Datenbank des

Geräte-Servers zur Verfügung gestellt. Die erforderlichen Bedieninteraktionen oder Ereignismeldungen des Supervisors werden dem Client über den Geräte-Server zur Verfügung gestellt.

Backup gewährleistet eine flexible Sicherheitsprüfung über Zugriffsberechtigungen auf Daten, Sicherungsgeräte und Sicherungsmedien. Ein Identifizierungs-Code auf dem Sicherungsmedium gewährleistet die Verwendung des richtigen Sicherungsmediums.

Ein Daten-Server generiert einen Datenstrom der Daten, die gesichert werden sollen. Sowohl Vollsicherungen als auch inkrementelle Sicherungen auf verschiedenen Stufen sind möglich. Der Daten-Server bietet die Möglichkeit, die Daten zu komprimieren. Darunter liegt eine Schnittstelle, die es erlaubt, den Datenstrom unterschiedlich aufzubereiten. Auch Nachfolgebehandlungen sind zugelassen.

Jedes Sicherungsgerät im Netzwerk hat seinen eigenen Supervisor. Er übernimmt die Steuerung des Geräts und unterstützt gerätespezifische Eigenschaften, so daß eine Schnittstelle unabhängig von dem spezifischen Gerät bedient werden kann. Zur Zeit gibt es Supervisoren für Magnetbandgeräte, Magnetbandkassettengeräte, Video-8-Systeme, Diskettenlaufwerke und Jukebox-Laufwerke mit optischen Platten.

Der Client repräsentiert die Schnittstelle zum Systemverwalter. Er ist verantwortlich für die Zugriffs- und Berechtigungskontrolle sowie für die Syntaxprüfung von Benutzerkommandos. Der Client kann in zwei verschiedenen Betriebsarten gestartet werden:

- Hintergrund
  Der Sicherungsauftrag wird in die Warteschlange gestellt. Dabei gibt es keinen Dialog mit dem Systemverwalter.
- Interaktiv
  Der Systemverwalter kann die laufenden Aufträge von seinem Terminal aus steuern.

Für alle Aktionen, die für die Steuerung und Konfiguration des Datensicherungswerkzeugs ausgeführt werden, stehen dem Benutzer drei verschiedene Möglichkeiten zur Verfügung:

- Shell-Kommandos
- FMLI-Menü für alphanumerische Terminals
- OSF/Motif-Schnittstelle

In die Struktur der Schnittstelle sind ausführliche Hilfefunktionen integriert. Eine mehrsprachige Unterstützung ist geplant.

Backup ist als Werkzeug für die Datensicherung in kommerziellen Netzwerken konzipiert. Es erlaubt die netzwerkweite Festlegung von Sicherungsbereichen und Sicherungszeiten und unterstützt eine große Anzahl verschiedener Sicherungsgeräte. Über das Medienmanagement stellt es sicher, daß gültige Datensicherungen nicht durch Fehler überschrieben werden. Backup entlastet den Systemverwalter bei seinen täglichen Sicherungsaufgaben.

Heterogenes Netzmanagement in SINIX: TRANSVIEW

Übersicht

In zunehmenden Maße werden in Unternehmen und Organisationen Netzwerke mit unterschiedlichen Rechnern eingesetzt. Für solche heterogenen Netze gibt es derzeit nur wenige integrierte Managementlösungen. Die Managementaufgaben in heutigen heterogenen Netzen müssen für jedes Teilnetz mit herstellerspezifischen Werkzeugen (Kommandosprachen, Verwaltungswerkzeuge) gelöst werden. Erforderlich ist ein übergreifendes Netzmanagement für heterogene Netzwerke mit einer einheitlichen Sicht auf alle Netzwerk- und Systemkomponenten, die zu steuern und zu überwachen sind. Voraussetzungen dafür sind:

- die Definition international standardisierter Protokolle und Dienste für den Austausch von Daten und Managementinformationen
- die Definition von Objekten, unabhängig von den Eigenschaften der Herstellersysteme
- die Definition von Managementeigenschaften

Dieser Bedarf wird in Zukunft steigen, vor allem mit der Verbreitung von verteilten Systemen, die sehr komplexe Interaktionen zwischen Netzknoten hervorrufen.

Die Familie der TRANSVIEW-Produkte von Siemens Nixdorf unterstützt die Netzwerksteuerung, die Überwachung und die Software-Verteilung. Aufbauend auf langjährigen Erfahrungen auf dem Gebiet des Netzmanagements und der Systemverwaltung hat Siemens Nixdorf mit TRANSVIEW ein modernes und flexibles Verwaltungssystem geschaffen, das die bestehenden OSI-Managementstandards und De-facto-Industriestandards (OSF/DME, SNMP, TCP/IP, SNA) unterstützt. TRANSVIEW-Produkte ermöglichen es, bestehende und zukünftige Systeme in homogenen und heterogenen Netzwerken effektiv zu unterstützen.

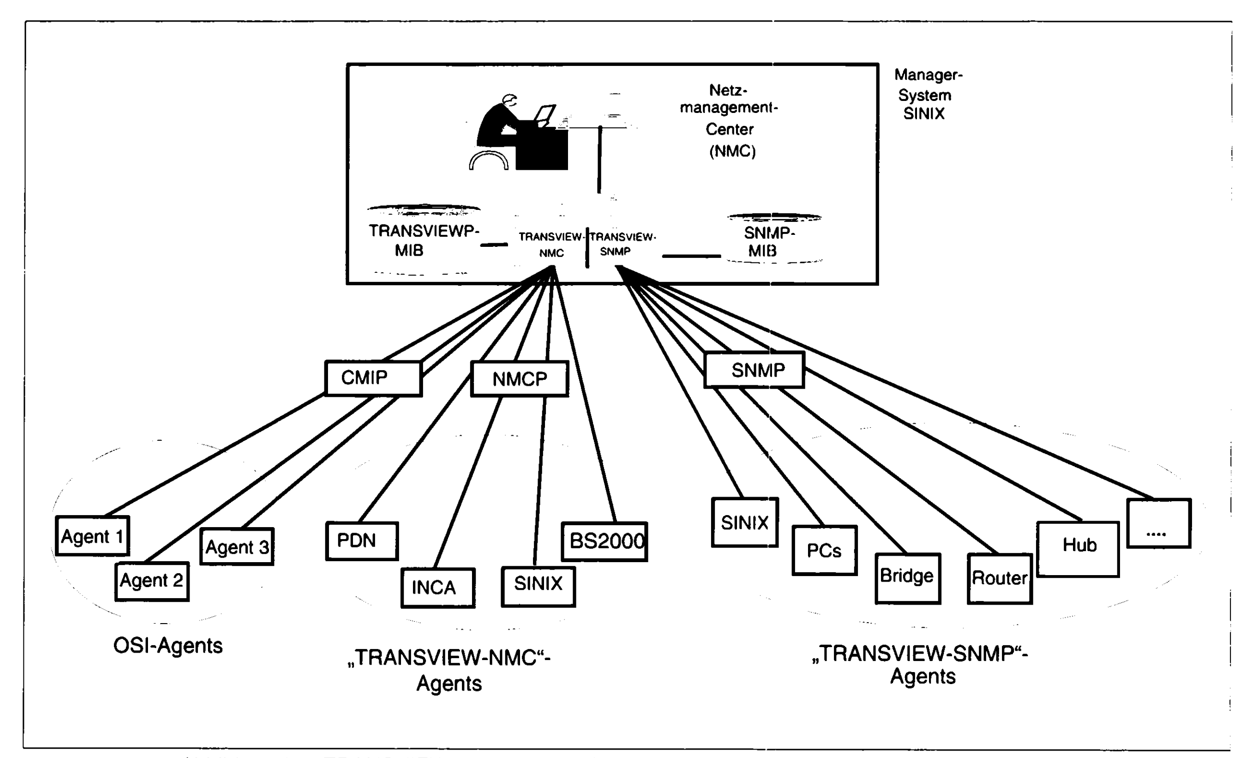


Abbildung 6-4: TRANSVIEW Manager und Agents

Die angebotenen Netzmanagement-Operationen umfassen Standard-Management-Anwendungen zur Netz- und Systemdefinition, Umkonfigurierung, Netz-, System- und Anwendungsüberwachung sowie die Ermittlung von Leistungsdaten. Die für TRANSVIEW geplanten Funktionen und Möglichkeiten werden in mehreren Stufen implementiert. In der ersten Stufe liegt der Schwerpunkt auf der Realisierung von Funktionen aus den Bereichen Netzüberwachung, Konfiguration, Fehlerbehandlung und Performance-Management.

Das TRANSVIEW-Netzmanagement basiert auf dem Manager-Agent-Prinzip. Es steuert und überwacht von einer zentralen Stelle aus eine Vielzahl unterschiedlicher Agents, die in Agent-Systemen lokalisiert sind.

Manager und Agent-Systeme kommunizieren auf der Basis folgender Management-Protokolle:

- NMCP (**N**etwork **M**anagement **C**ommunication **P**rotocol), einer Erweiterung des vorhandenen S3-Protokolls für den homogenen Betrieb mit Siemens-Nixdorf-Systemen (SINIX-, BS2000-, PDN- und INCA-Systemen).
- CMIP (**C**ommon **M**anagement **I**nformation **P**rotocol) zur Unterstützung heterogener Netze auf der Basis von OSI-Managementstandards.
- SNMP (**S**imple **N**etwork **M**anagement **P**rotocol) zur Unterstützung TCP/IP-Netzen.

TRANSVIEW führt die Übertragung von Management-Informationen unabhängig vom Transportsystem durch. Unterstützt werden z.B. NEA, ISO, TCP/IP und SNA-Netze.

Zu TRANSVIEW gehören folgende Produkte für Netz-, System- und Anwendungsmanagement:

- TRANSVIEW-NMC
- TRANSVIEW-SNMP
- TRANSVIEW-DSM
- TRANSVIEW-DAME

Zusätzlich zu diesen Produkten gehören auch die TRANSDATA-Netzmanagement-Produkte zur TRANSVIEW-Produktpalette. Besonderes Augenmerk wurde darauf verwendet, existierende und erprobte Produkte in das TRANSVIEW-Konzept einzubinden, die harmonisch mit den neuen Produkten zusammenarbeiten. Mit TRANSIT-NVS (**N**et**V**iew **S**upport) werden SINIX-Systeme in das SNA-Management eingebunden (über Entry Point und Service Point). TRANSVIEW-NMC und TRANSVIEW-SNMP werden in Zukunft mit NetView über TRANSIT-NVS kommunizieren.

TRANSVIEW wird ständig weiterentwickelt, um dem Bedarf des Marktes entgegenzukommen. Ein wichtiges Ziel ist die Integration funktionaler Managementelemente, die bislang auf eigenen Plattformen liefen. Die nächsten Versionen von TRANSVIEW unterstützen weitere Agent-Systeme von Siemens Nixdorf und neue Managementfunktionen, wie z.B. *license*. Schritt für Schritt wird TRANSVIEW die OSF/DME-Spezifikationen zur Vereinheitlichung von System- und Netzmanagement verwirklichen. Dies trifft teilweise heute schon für DME-spezifizierte Schnittstellen zu.

TRANSVIEW-NMC

TRANSVIEW-NMC ist das Netzmanagement-Produkt von Siemens Nixdorf, das im **N**etwork **M**anagement **C**enter (NMC) eingesetzt wird. Die zentrale Komponente von TRANSVIEW-NMC ist die offene Plattform. Sie ist das integrierende Element für alle Anwendungen, mit denen die umfassenden Funktionen von TRANSVIEW-NMC realisiert werden. Der Zugang besteht über das TRANSVIEW-API INMS (**I**nterface **N**etwork **M**anagement **S**ervice), die Schnittstelle zu den Netzmanagement-Diensten und dem X/Open XMP-API (**X**/Open **M**anagement **P**rotocol API). TRANSVIEW berücksichtigt die wichtigsten Merkmale der auf OSI basierenden offenen Managementarchitektur. Das NMC ist ein SINIX-System, das als Manager verschiedene Agent-Systeme steuert und überwacht. Es steht eine konfigurierbare, objektorientierte Bedienoberfläche zur Verfügung, die auf dem X Window System und SINIX/windows basiert.

Die wesentlichen Bestandteile des NMC sind:

- der Kernel (zentrales Element der TRANSVIEW-Plattform)
- die Netzbausteine für die Kommunikation von Management-Informationen
- die grafische Bedienoberfläche
- die Standard- oder betreibereigenen Netzmanagement-Anwendungen
- die zentrale Datenbank
- das Meldungswesen

Die meisten Funktionen von TRANSVIEW-NMC sind über Netzmanagement-Anwendungen verfügbar:

- Network Monitor
- Performance Monitor
- Session Monitor
- Netzdefinition
- Netzkonfiguration für Systeme

Die Standardfunktionen können durch betreibereigene Funktionen ergänzt werden, um z.B. das Netzmanagement-System auf die Bedürfnisse des Betreibers zuzuschneiden. Über das objektorientierte, offene Management-API INMS (**I**nterface **N**etwork **M**anagement **S**ervice) kann auf die zentrale Datenbank zugegriffen werden. Die Basis-Datenbanksysteme sind das **E**nterprise **R**epository **M**anagement **S**ystem (ERMS) und das Datenbanksystem INFORMIX.

TRANSVIEW unterstützt in der ersten Stufe durch folgende Netzmanagement-Funktionen die Systeme anderer Hersteller, die mit dem NMC über das standardisierte OSI NM-Protokoll CMIP (**C**ommon **M**anagement **I**nformation **P**rotocol) kommunizieren können:

- Network Monitor
- Netzdefinition
- grafische Bedienoberfläche

Dabei können Objekte angesprochen werden, die von ISO oder dem NM-Forum festgelegt wurden. Außerdem wendet TRANSVIEW die von EWOS (**E**uropean **W**orkshop for **O**pen **S**ystems) festgelegten Profildefinitionen für das **M**anagement **I**nformation **P**rotocol MIP (AOM12) an.

Globale Struktur

Die Architektur, die Informationsstruktur und die wichtigen internen Mechanismen sind wesentlich geprägt durch die OSI-Netzmanagement-Standards von ISO und CCITT sowie durch die Arbeiten des NM-Forums. Ein Beispiel dafür ist die Objektorientierung, die sich in der Benutzerschnittstelle (Programmierung und Bedienschnittstelle), in der internen Struktur und in der Objektbibliothek zur Beschreibung eines TRANSDATA-Netzes manifestiert. Die Objektbibliothek ist grundlegend für die konsistente Datenhaltung, deren Konzept sich an den OSI-Prinzipien der **M**anagement **I**nformation **B**ase (MIB) zum Zugriff auf Management-Informationen orientiert. Die Struktur der Management-Informationen richtet sich nach den von ISO festgelegten Regeln (**S**tructure of **M**anagement **I**nformation - SMI). Die von TRANSVIEW eingeführten Dienste entsprechen den bei ISO definierten CMIS-Diensten. Das von ISO definierte Verfahren für die Verteilung von Meldungen wird auch für die internen systemspezifischen Meldungen verwendet. Die Architektur von TRANSVIEW stimmt so im wesentlichen mit der von ISO definierten Netzmanagement-Architektur überein und legt die Regeln für die Struktur des TRANSVIEW Management-Systems (Plattform, APIs, Module) fest.

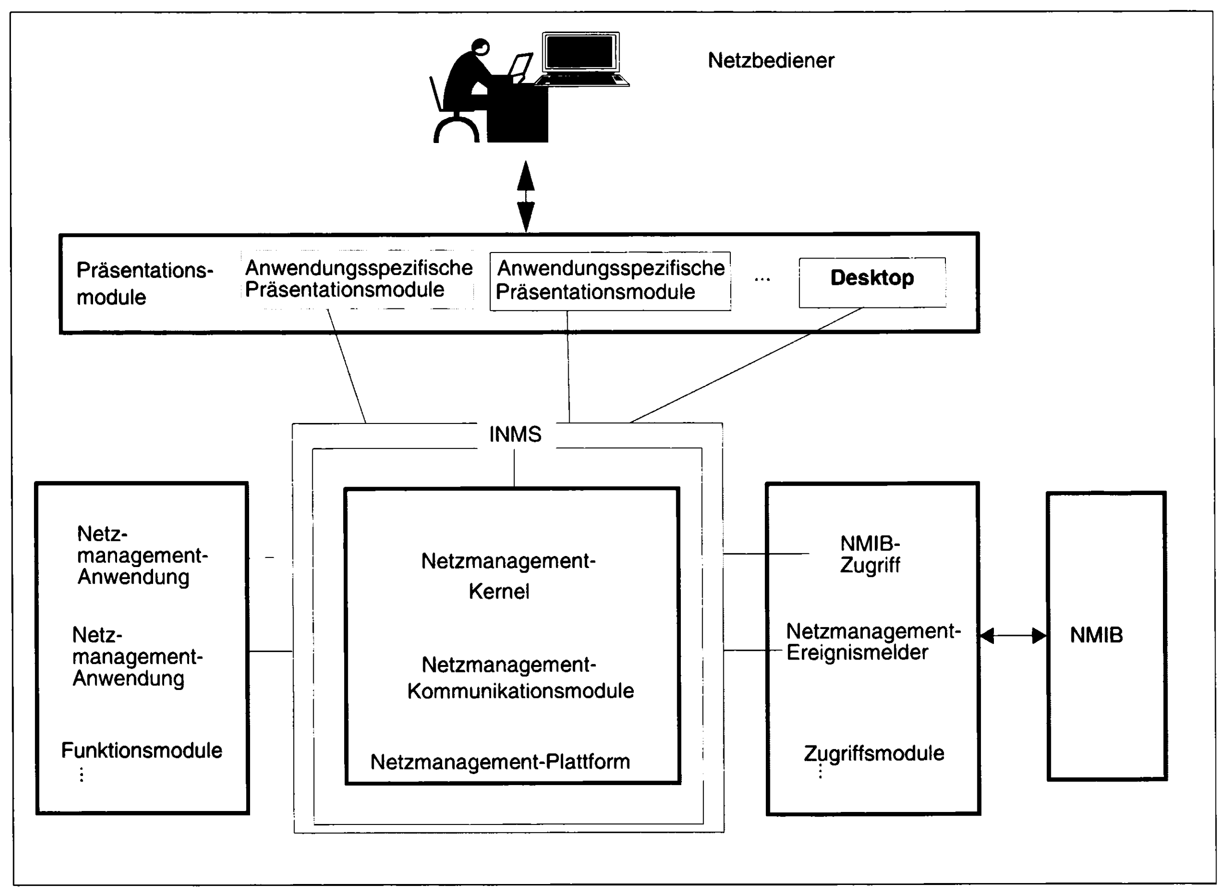


Abbildung 6-5: Die Netzmanagement-Plattform

In TRANSVIEW führen die **N**etwork **M**anagement **S**ervices (NMS) die Managementfunktionen aus. Diese Dienste umfassen Empfang, Übertragung und Zustellung der Netzmanagement-Aufträge, der Auftragsergebnisse und Meldungen der Netzmanagement-Ereignisse. Dienstnutzer sind die Manager und Agents von TRANSVIEW. Zum Informationsaustausch zwischen Managern und Agents, die auf unterschiedlichen Systemen lokalisiert sind, verwendet TRANSVIEW die Netzkommunikationsdienste. Der Informationsaustausch erfolgt über Netzmanagement-Protokolle. Die mit NMS angebotenen Dienste stehen an der Schnittstelle INMS (**I**nterface to **N**etwork **M**anagement **S**ervices) zur Verfügung. Die INMS ist eine offene Anwendungs-Programmierschnittstelle (API). Sie kann von Standard-Netzmanagement-Anwendungen, die mit dem Produkt TRANSVIEW-NMC zur

Verfügung gestellt werden, wie auch von betreibereigenen Anwendungen benutzt werden. Die TRANSDATA-Objektbibliothek ist ein integraler Bestandteil von TRANSVIEW und wird zusammen mit TRANSVIEW ausgeliefert. Es stehen Werkzeuge zur Verfügung, mit denen sie erweitert werden kann.

Im TRANSVIEW-NMC sind die Netzmanagement-Dienste als die Netzmanagement-Plattform implementiert. Die Plattform besteht aus dem Netzmanagement-Kernel und den Netzmanagement-Kommunikationsmodulen (Netzbausteine). Die Aufgaben des Kernels umfassen das Protokollieren und Zustellen von Aufträgen, Auftragsergebnissen und Ereignismeldungen an den jeweiligen Empfänger sowie die Berechtigungsprüfung. Die Netzbausteine ermöglichen den Zugang zu den Objekten oder zu Netzmanagement-Informationen in entfernten Agent-Systemen durch die Nutzung von Netzmanagement-Protokollen. TRANSVIEW unterstützt verschiedene Agent-Systeme. Die mit der Plattform gebotenen Dienste stehen an der Programmschnittstelle INMS (**I**nterface **N**etwork **M**anagement **S**ervice) zur Verfügung. Die nachfolgend aufgeführten Module nutzen diese Dienste zum Einholen von Management-Informationen.

Präsentationsmodule

Diese umfassen den Baustein *Desktop* und anwendungsspezifische Präsentationsbausteine. Der Baustein *Desktop* ist die Arbeitsumgebung für den Netzbediener (z.B. grafische Darstellung der Netzkonfiguration, Anzeigen von Ereignissen im Netz, Verwaltung von Auftragsfenstern).

Funktionsmodule

Die Funktionsmodule sind Netzmanagement-Anwendungen. Sie bieten dem Netzbetreiber vielfältige Funktionen zur Steuerung und Überwachung der System- und Netzkomponenten, wie z.B. Auswertung und Aufbereitung von Netzmanagement-Informationen. TRANSVIEW stellt hier Standard-Anwendungen für allgemein nutzbare Einsatzfälle zur Verfügung, wie z.B. Netzdefinition, Netz-Monitor, Verbindungs-Monitor.

Zugriffsmodule

Die Zugriffsmodule ermöglichen den Zugang zu den Netzressourcen und zu den Netzmanagement-Informationen. Bei Agent-Systemen sind dies die lokalen Netzressourcen, am NMC sind zusätzlich die in der zentralen Datenbank gespeicherten und von den Netzmanagement-Anwendungen ermittelten Informationen verfügbar.

TRANSVIEW-SNMP

Neben den bei ISO definierten OSI-Management-Standards wurde im Rahmen der Internet Community mit SNMP (**S**imple **N**etwork **M**anagement **P**rotocol) ein weiterer Netzmanagement-Standard festgelegt. SNMP kann in heterogenen LANs (Ethernet, Token Ring, FDDI) eingesetzt werden, die mit dem Protokoll TCP/IP betrieben werden. Das SNMP-Management basiert wie das OSI-Management auf dem Manager-Agent-Prinzip. Die wesentlichen Komponenten des SNMP-Managements sind Manager, Agent und Management Information Base (MIB). Bei Siemens Nixdorf steht als Produkt, mit der die Management-Rolle realisiert wird, das TRANSVIEW-SNMP zur Verfügung. TRANSVIEW-SNMP läuft auf SINIX-Systemen und ist als erstes Produkt der TRANSVIEW-Familie auf dem Markt erschienen. Es unterstützt eine Vielzahl von SNMP-Agents, wie z.B. Bridges, Router, Gateways, Hubs, SINIX-Systeme und BS2000-LAN-Kanaladaptoren. Alle aktuellen SINIX-Systeme verfügen über SNMP-Agents. Sie unterstützen die komplette standardisierte MIB-II und sind über eine Programmschnittstelle erweiterbar für weitere standardisierte und private MIBs.

TRANSVIEW-SNMP (Manager)

TRANSVIEW-SNMP ist das Produkt von Siemens Nixdorf, mit dem die Management-Funktionalität realisiert wird. Die wesentlichen Elemente von TRANSVIEW-SNMP sind die SNMP-Protokollmaschine, eine Plattform, auf der Management-Anwendungen, wie z.B. Objekt-Management, Alarm-Management usw. aufsetzen können, und eine X Window und OSF/Motif basierte, komfortable Bedienoberfläche. Bei dieser Struktur wurden die wichtigsten Merkmale einer offenen Management-Architektur berücksichtigt, wie sie auch bei ISO für das OSI-Management definiert wurde.

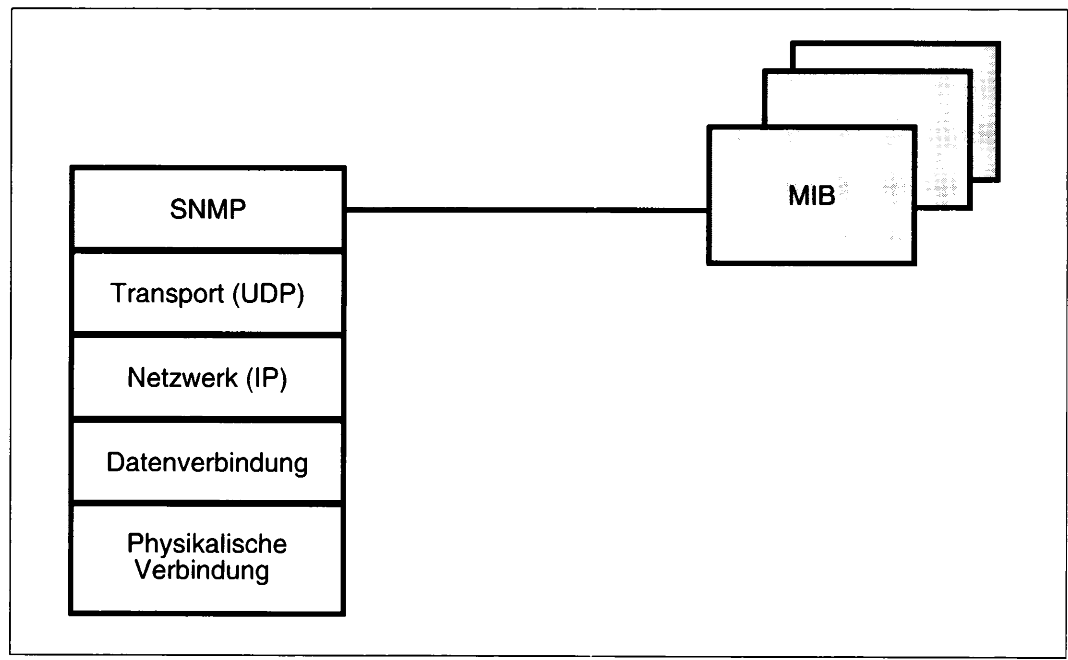


Abbildung 6-6: Das SNMP-Protokoll.

Über vier Benutzerschnittstellen hat der Betreiber Zugang zur TRANSVIEW-SNMP-Plattform und damit die Möglichkeit, sein Management-System optimal an seine Bedürfnisse anzupassen:

- Das Anwendungs-API zum Anschließen von Standard- und betreibereigenen Management-Anwendungen.
- Das API für den fernen Zugriff mit Zugriffsfunktionen auf Objekte in entfernten Agent-Systemen und zum Empfangen von Meldungen dieser Systeme. Über dieses API können neben SNMP weitere Protokollstacks, wie z.B. für CMIP oder für betreibereigene Protokolle, angesprochen werden.
- Das Bedienoberflächen-API zur Gestaltung des Bildschirm-Layouts (Farben, Icons, Vordergrund, Hintergrund usw.)
- Das Alarm-Management-API zur automatischen Reaktion auf Fehlerzustände.

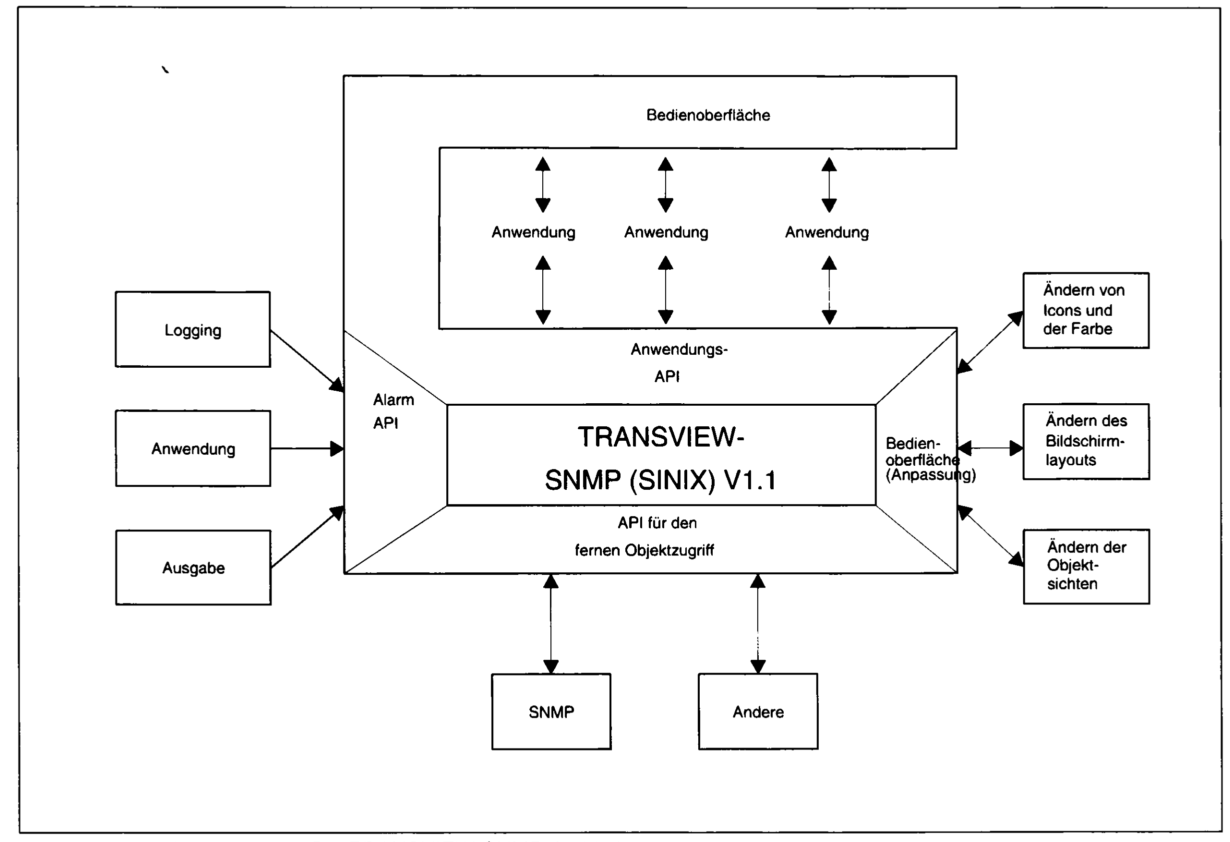


Abbildung 6-7: Die TRANSVIEW-SNMP-Architektur.

Funktionen von TRANSVIEW-SNMP

TRANSVIEW-SNMP ist über eine grafische Bedienoberfläche auf der Basis von X Window und OSF/Motif zu bedienen. Diese Bedienoberfläche kann der Betreiber leicht selbst gestalten. Agents können über Icons aus der Icon-Bibliothek oder selbst erstellte Icons dargestellt werden. Für einen Agenttyp sind ein oder mehrere Icons verwendbar. Darüber hinaus können mit der Discovery-Funktion Netzkonfigurationen automatisch erfaßt und verglichen sowie unbefugtes Eindringen festgestellt werden.

Die wichtigsten Funktionen von TRANSVIEW-SNMP sind:

- Regelbasierte Fehlerbehandlung (Polling mit einstellbaren Intervallen, Fehlergewichtung, Zustandsübergangsdiagramme für automatische Reaktionen, Anzeigen im Netz-Monitor, erweiterbare Management Information Base (private MIB)
- Konfigurations-Management (Network Monitor zur Zustandsüberwachung, Definition von Netzen)
- Performance-Management (Erzeugung von Auslastungsdaten als Grafiken, Listen oder Tabellen, Protokoll-Management)
- Manager-Kaskadierung
- Unterstützung des Produkts NetProbe
- Bridge-Management
- SQL-Schnittstelle zum Datenbanksystem INFORMIX

SNMP bietet als Standard eine Zugriffskontrolle über die sogenannte „community", die mit einem Paßwort vergleichbar ist. Zusätzlich ist der SINIX-Zugriffs- und Berechtigungsschutz gewährleistet. Die weitere Entwicklung von TRANSVIEW-SNMP umfaßt schwerpunktmäßig den Anschluß von NetView, komfortables Konfigurieren und Steuern von Hubs sowie das komfortable Erstellen von Statistiken.

TRANSVIEW-SNMP läuft sowohl eigenständig als auch im Verbund mit TRANSVIEW-NMC. Im Verbundbetrieb wird TRANSVIEW-SNMP vom TRANSVIEW-NMC aus gestartet und beendet. Seine Oberfläche ist in die von TRANSVIEW-NMC integriert. Außerdem können Alarmmeldungen von TRANSVIEW-SNMP an TRANSVIEW-NMC weitergeleitet werden, so daß das gesamte Netz von einem Terminal aus überwacht und gesteuert werden kann.

SNMP-Agents auf SINIX-Systemen

SNMP-Agents wickeln einerseits die Kommunikation über SNMP ab und greifen andererseits lesend und schreibend auf die Objekte ihrer Netzkomponenten zu. Netzkomponenten sind z.B. Bridges, Router, Hubs, aber auch SINIX-Systeme. Durch die Definition privater MIBs ist SNMP für Management-Aufgaben in beliebigen Bereichen einsetzbar. Die getrennte Festlegung der Standards für das SNMP-Protokoll und die MIB hat auch die Software-Architektur des Siemens Nixdorf-Produktes SINIX-SNMP-Agent geprägt: SNMP und MIB-II sind dabei in getrennten Modulen implementiert (Core Agent und Agent Extension), die über eine Programmschnittstelle zusammenarbeiten. Über diese Programmschnittstelle können auch weitere MIB-Implementierungen angeschlossen werden.

Der SINIX-Agent unterstützt die Standard-MIB-II. Der SNMP-Agent V1.0 ermöglicht zunächst den lesenden Zugriff auf eine Untermenge der in der MIB-II definierten Objekte. Eine Erweiterung, im SNMP-Agent V2.0 realisiert, unterstützt alle Objekte mit lesendem und schreibendem Zugriff entsprechend den Festlegungen des MIB-II-Standards. Die Schnittstelle zwischen Core Agent und Agent Extension ist so angelegt, daß Agent Extensions dynamisch dazugebunden werden können. Dies hat den Vorteil, daß Agent Extensions entkoppelt vom Core Agent freigegeben werden können. Ein Siemens-Nixdorf-Produkt oder eine Anwendung, die über das SNMP-Management gesteuert und überwacht werden soll, enthält eine Agent Extension, die als Bestandteil dieses Produkts ausgeliefert wird. Bei der Installation des Produkts wird seine Agent Extension automatisch an den bereits auf dem SINIX-System vorhandenen Agent angebunden und ist damit von einem Manager über das SNMP-Protokoll ansprechbar.

Für die Definition von herstellerspezifischen MIBs hat sich Siemens Nixdorf von der **I**nternet **A**ssigned **N**umbers **A**uthority (IANA) einen Knoten im Baum der weltweit eindeutigen Objektnamen zuweisen lassen. Unter diesem Knoten werden die herstellerspezifischen Objekte angesiedelt. Siemens Nixdorf stellt auch seinen Kunden unter dieser Nummer einen Unterbaum zur Definition von deren privater MIB zur Verfügung.

TRANSVIEW-DSM

TRANSVIEW-DSM (**D**istributed **S**ystems **M**anagement) umfaßt Funktionen zur Software-Verteilung, Software-Verwaltung und Software-Installation. Es kann auf einer Vielzahl von Betriebssystemen, wie z.B. SINIX, MS-DOS, MS-Windows, und OS/2, Software verteilen, installieren und verwalten. Das kann Software von Siemens-Nixdorf-Produkten sein, Software von Fremdherstellern oder Kunden-Software. Unter DSM dient ein Rechner im Netz als sog. MS (Managing System), das alle Informationen speichert und die Verwaltung durchführt.

Der herausragende Vorteil von TRANSVIEW-DSM ist, daß geschulte und erfahrene System- und Netzwerkverwalter von einem Rechner aus Software installieren, verwalten und aktualisieren können, und Benutzer an anderen Systemen oder anderen Rechnern im Netz von der Durchführung dieser Aufgaben nicht betroffen sind. Diese können sich auf ihre Arbeit konzentrieren, was zur allgemeinen Produktivitätssteigerung beiträgt.

TRANSVIEW-DAME

Das Produkt TRANSVIEW-DAME (**D**ynamic **A**dministration **M**anagement **E**nvironment) steuert und überwacht Systeme und deren Anwendungen, z.B. SINIX, BS2000, UNIX, MS-DOS, OS/2-Systeme. Zu den Standard-Funktionen gehören Ausfallüberwachung, Domänen-Konzept und automatische Ereignisüberwachung.

Sichere Systemverwaltung

Die wachsende Anzahl vernetzter Systeme und ihre Kommunikation über LAN und WAN haben zu Entwicklungen geführt, die die Sicherheit in Systemen und Netzwerken betreffen. Gleich welche Sicherheitsmaßnahmen eingeführt sind, sie liegen in der Verantwortung der System- und Netzwerkverwalter. Für diese Aufgaben brauchen sie entsprechende Werkzeuge. Siemens Nixdorf und andere Unternehmen, die sich mit offenen Systemen beschäftigen, arbeiten an der Entwicklung von Standards und Protokollen für die sichere Systemverwaltung. Die Entwicklung von Standards und Produkten für die Sicherheit ist im Abschnitt „Erweiterte Sicherheitsfunktionen unter SINIX" auf Seite 191 beschrieben. Eine Implementierung von Software für Netzsicherheit, der Sicherheitdienst von DCE, ist in Abschnitt „Verteilte Verarbeitung: DCE als Lösungsmodell" auf Seite 132 beschrieben.

7 Verteilte Verarbeitung

Überblick

Moderne Firmen folgen dem Trend der Zeit und arbeiten heute nicht mehr nur mit großen Rechenzentren, sondern bevorzugen dezentrale Netzwerklösungen. Anwender wollen auf gemeinsame Ressourcen zugreifen, seien es Datenbanken oder Peripheriegeräte (wie Hochleistungs-Laserdrucker oder optische Laufwerke). Dies kann am besten durch die gemeinsam nutzbare Rechenleistung von PCs und Abteilungsrechnern erreicht werden. Dafür steht nun ein breites Spektrum von Anwendungen zur Verfügung, die alle den gleichen Lösungsansatz haben: verteilte Verarbeitung. Die umfassendste und effektivste Anwendung in diesem Bereich, mit den meisten Funktionen und Werkzeugen, ist DCE (**D**istributed **C**omputing **E**nvironment).

Ziel der verteilten Verarbeitung ist es, allen Benutzern die in einem Netzwerk bereit gehaltenen Ressourcen verfügbar zu machen, um so eine leistungsfähige Arbeitsumgebung zu schaffen. Verteilte Verarbeitung ist deshalb vor allem für solche Anwender interessant, die schon heute über komplexe Rechnersysteme in einem hochentwickelten Netzverbund verfügen. In Kürze wird die verteilte Verarbeitung standardmäßig Teil der Datenverarbeitung im Geschäftsleben sein.

Anbieter von Betriebssystemen, Netzwerken und unabhängige Softwarehäuser haben in der Vergangenheit im Bereich der verteilten Verarbeitung eine breite Palette von Produkten angeboten. Diese Produkte ermöglichen bis zu einem gewissen Grad die Kommunikation zwischen Rechnern. Mit diesen Produkten kann man E-Mail verschicken, sich Drucker teilen, Dateien austauschen, sich an fernen Rechnern anmelden oder Speicherplatz auf Festplatten teilen.

Heute suchen Kunden nach einer Lösung, die ihnen die Vorteile der integrierten und offenen Systeme bringt, damit alle schon vorhandenen vernetzten Systeme miteinander kommunizieren können, ob dies nun Großrechner, Minicomputer, Work-

stations oder PCs sind. Eine umfassende Systemlösung mit verteilter Verarbeitung ist offensichtlich wünschenswert, um die Bedürfnisse eines Betriebs im Bereich der Informationssysteme zu erfüllen.

Was Anwender nicht Fall wünschen, ist verteilte Verarbeitung, die nur auf einer bestimmten Rechnerarchitektur eines einzigen Anbieters basiert. Gewünscht wird vielmehr echte, wirkungsvolle Zusammenarbeit. Gefragt ist verteilte Verarbeitung, die sich ohne weiteres in einen heterogenen Maschinenpark mit unterschiedlichen Architekturen, Betriebssystemen und Kommunikationsprotokollen einpassen läßt.

Das oberste Ziel der verteilten Verarbeitung ist es, Ressourcen und Dienste netzweit, also von jedem Rechner aus, in einer Firma nutzen zu können. Die beste Lösung für dieses Problem ist die Client-Server-Architektur.

Client-Server-Architektur

Während man früher von einem Konzept ausging, das eine Anwendung als etwas Monolithisches ansah, geht man in der Client-Server-Architektur davon aus, daß Komponenten einer Anwendung auf Clients und Server verteilt werden können. Die Serverkomponenten stellen dabei Ressourcen allgemein zur Verfügung. Die Clientkomponenten vermitteln lokal den Anwendungen die Dienstleistungen der Server.

Theoretisch ist es möglich, daß die Serverkomponente einer Anwendung von einem einzigen Rechner aus alle Netzwerknutzer bedienen kann. Die Praxis sieht allerdings meist so aus, daß ein Server auf mehreren Rechnern im Netz liegen kann und so durch die Verringerung der Leitungswege im Netz die Verfügbarkeit deutlich erhöht wird. Bekannteste Beispiele für diese Architektur sind die häufig eingesetzten Drucker-Server, File-Server oder Mail-Server.

Die Clientkomponente einer Anwendung wird jeweils auf dem Rechner installiert, auf dem die betreffende Anwendung von Benutzern gebraucht wird. Die Clientkomponente arbeitet als Schnittstelle zum Server, d.h. sie steuert sowohl den Datenfluß im Netzwerk zwischen Anwender und Server als auch die Daten, die auf dem Bildschirm ausgegeben werden. Eine Ausnahme bildet X Window; hier steuert der Server die Bildschirmausgabe, während die Clients beliebige Anwendungen auf einem beliebigen Rechner im Netz sein können, die dann auf diese Bildschirmsteuerung zurückgreifen.

Schon in der älteren Literatur findet man den Begriff Client-Server-Architektur. Damit ist jedoch nur ein einfacher Verbund von Client-Rechnern und Server-Rechnern gemeint. In diesem Modell kommen nur dedizierte Client- oder Server-Systeme zum Einsatz, die entweder Träger von Client- oder Träger von Serverkomponenten sind. Dies ist beispielsweise in einem Netzverbund aus PC-Clients und SINIX-Servern gegeben. In Systemen mit Client-Server-Anwendungen mit komplizierter Infrastruktur (z.B. DCE) unterscheidet man nicht strikt zwischen einem dedizierten Client- oder Server-Rechner; ein Rechner kann gleichzeitig Träger sowohl von Clientkomponenten als auch von Serverkomponenten sein.

Im komplexen Client-Server-Modell ist die Rolle der Clients und Server relativ und dynamisch. Was darunter zu verstehen ist, soll an folgendem Beispiel erläutert werden: Auf einem Rechner befindet sich der Drucker-Server. Ein anderer Rechner arbeitet als File-Server. Der erste Rechner ist damit jedoch zugleich auch Client, denn er greift auf Dateien des File-Servers zu. Daneben können Serverfunktionen auch verteilt werden. Das bedeutet z.B., daß der Druckauftrag eines Clients auf einem er-

sten Rechner an den Drucker-Server eines zweiten Rechners weitergegeben wird, während ein anderer Druckauftrag des ersten Rechners vom Drucker-Server eines dritten Rechners bedient wird. Ein Server kann mehrere Dienste zur Verfügung stellen, so kann z.B. der Server einer Kundenkreditbank mehrere Dienstleitungen wie Kontostandsabfrage, Barabhebung, Einzahlung oder Überweisung anbieten.

Die zentrale Aufgabe bei der Konzipierung von verteilten Anwendungen besteht darin, die Anwendungen so zu strukturieren, daß eine sinnvolle Aufgabenteilung im Netz erreicht wird. Dies erfordert die Festlegung von Funktionseinheiten, d.h. Client- und Serverkomponenten, und die optimale Verteilung von Anwendungsfunktionen, Diensten und Daten zwischen Client und Server. Dabei sind folgende Kriterien zu berücksichtigen:

- Netztopologie
- Verfügbarkeit von Systemdiensten
- Schnittstellen
- Kosten

Eine weitere wichtige Aufgabe neben der Festlegung von Client- und Serverkomponenten ist die Steuerung der Zusammenarbeit zwischen diesen Komponenten. Diese kann einfacher Art sein (1:1 Beziehung zwischen Server und Client), kann aber auch komplexer sein (z.B Zugriff eines Clients auf mehrere Server während einer Transaktion). Schließlich gehört dazu, daß die Client- und Serverkomponenten eine optimale Leistung durch die im Multitasking/Multiusing gegebenen Möglichkeiten der Prozeßparallelisierung bzw. asynchronen Verarbeitung erreichen.

Dienste der verteilten Verarbeitung

Die ersten Produkte im Bereich der verteilten Verarbeitung bieten für besondere Aufgaben optimale Lösungen. Im folgenden Abschnitt werden drei dieser Dienste der verteilten Verarbeitung beschrieben: NFS, LM/X und SPOOL.

Das verteilte Dateisystem NFS

Das **N**etwork **F**ile **S**ystem (NFS) wurde ursprünglich von Sun Microsystems entwickelt, wurde aber bald auch von einer großen Zahl weiterer Softwarehäuser angeboten, darunter auch von Siemens Nixdorf. Das Network File System von Siemens Nixdorf heißt **D**istributed **F**ile **S**ystem (DFS).

In einem späteren Abschnitt dieses Kapitels wird das DFS von DCE beschrieben (siehe Abschnitt „DCE-Komponenten" auf Seite 135). Dieses DFS verfügt über wesentlich bessere Leistungsmerkmale als NFS und ist außerdem in DCE eingebunden, hat also damit eine vollständige Laufzeitumgebung für Entwicklung und Anwendung von Programmen in verteilter Arbeitsumgebung.

NFS, eine Technologie, die schon seit geraumer Zeit verfügbar ist, stellt alle Grundfunktionen für ein verteiltes Dateisystem zur Verfügung. Dieses Dateisystem kann innerhalb eines lokalen Netzverbundes verteilt sein und von allen NFS-Client-Systemen innerhalb dieses Netzes erreicht werden. NFS erlaubt Anwendern, mittels Programmen auf Client-Rechnern via Netzwerk Lese- und Schreibzugriff auf Dateien eines Server-Rechners. Der Zugriff auf eine ferne Datei geschieht transparent für das Client-Programm und den Anwender. Das heißt, die Dateioperationen für die lokalen und die fernen Dateisysteme sind identisch. Ein Programm, das auf dem Client-Rechner abläuft, bemerkt also nicht, ob Dateioperationen an einer lokalen oder an einer fernen Datei ausgeführt werden. Soll eine Dateioperation von einer lokalen auf eine ferne Datei umgelenkt werden, so ist dafür immer eine Operation des lokalen Betriebssystemkerns notwendig.

Ein weiterer Aspekt des transparenten Zugriffs ist die Geschwindigkeit, mit der auf Daten innerhalb des Netzes zugegriffen werden kann. Der Zugriff auf Daten muß so schnell erfolgen, daß es nur einen kaum wahrnehmbaren Unterschied zwischen dem Zugriff auf eine NFS-Ressource und einem lokalen Plattenzugriff gibt. Es sollten 80% des Datendurchsatzes einer lokalen Platte erreicht werden. Oft erzielt der Zugriff über NFS auf einen dedizierten NFS-Server einen besseren Datendurchsatz, als dies mit einer langsameren lokalen Platte möglich ist.

NFS war ursprünglich nur auf UNIX-Systemen verfügbar. Aufgrund der besonderen Nutzbarkeit wurde es bald auf eine Vielzahl von Systemen portiert, vom Großrechner bis zum PC für die unterschiedlichsten Betriebssysteme. Siemens Nixdorf liefert DFS (NFS) für SINIX-Systeme, BS2000- Systeme (Server) und PCs (Clients).

Integration von SINIX- und PC-Netzen mit LM/X

Ursprünglich waren PCs nur dazu gedacht, die für einen Arbeitsplatz typischen PC-Anwendungen, Laufwerke und Drucker bereitzustellen. Aufgrund der Möglichkeiten, die PCs heute bieten (Rechnerleistung, Speicherkapazität, grafische Bedienoberfläche, individuelle Einstellung der persönlichen Arbeitsumgebung, Verfügbarkeit usw.) sowie ihrer steigenden Verbreitung, ist es durchaus sinnvoll, PCs in bestehende IT-Lösungen der Unternehmen zu integrieren bzw. neue IT-Konzepte unter Einbeziehung von PCs zu realisieren.

Grundsätzlich kann man zwischen drei Integrationsebenen unterscheiden, wobei die Grenzen zwischen diesen Ebenen fließend sind:

- PCs können dazu verwendet werden, Terminals zu emulieren (z.B. X-Terminals) und ermöglichen damit den Zugriff auf die Ressourcen anderer Computer (z.B. einen Mehrplatzrechner oder einen Großrechner).
- PCs können an ein LAN angeschlossen werden und sich auf diese Weise Ressourcen wie Drucker oder zentral auf Platte gehaltene Daten teilen.
- PCs können als Plattform für Client-Server-Anwendungen genutzt werden, sobald ein Mehrplatzrechner in das LAN eingebunden wird. Dieser stellt dann anderen Rechnern Ressourcen, wie z.B. eine große Festplatte, zur Verfügung. Geht man darüber hinaus, stößt man in Bereiche, die den echten verteilten Systemen zuzurechnen sind.

Mit Erscheinen der MS-DOS-Version 3.0 im Jahr 1984 wurde es möglich, PCs mit MS-DOS als Betriebssystem unter Verwendung des LAN-Managers (LM) in lokale Netzwerke (LAN) einzubinden. Der LAN-Manager bietet PCs transparenten Zugriff auf netzweit zugängliche Drucker und Plattenressourcen sowie eine Schnittstelle (API), welche die Kommunikation zwischen einzelnen Programmen ermöglicht.

Der LAN-Manager wurde von Microsoft als Netzwerkbetriebssystem für PC-LANs entwickelt, ist aber mittlerweile auch auf SINIX-Systemen unter der Bezeichnung LM/X verfügbar. Der LAN-Manager ist auf unterschiedlichen Transportsystemen ablauffähig, was für die Anwendungen aber immer transparent bleibt. Der LAN-Manager besteht aus einer Client-Komponente, die auf den Client-Systemen MS-DOS, MS-Windows und MS-Windows NT eingesetzt wird, und aus der auf dem SINIX-System installierten Server-Komponente (LM/X).

Für die Realisierung verteilter Anwendungen unter dem LAN-Manager stehen die folgenden Dienste und Schnittstellen, im folgenden nach Umfang und Zugriffsrechten differenziert, zur Verfügung:

- gemeinsamer Zugriff auf Dateien (shared files)
  - gemeinsamer, transparenter Zugriff auf Dateien des Servers
  - File- und Record-Locking
- gemeinsame Druckerbenutzung (shared print)
  - gemeinsame Ausgabe auf Drucker am Server
  - Spoolsystem für Pufferung und Druckausgabe
- Benutzung von asynchronen Geräten des Servers (shared device))
  - gemeinsamer, sequentieller Zugriff
  - Unterstützung aller SINIX-SPOOL-Systeme
- Named Pipes des LAN-Managers zur Unterstützung der bidirektionalen Programm-Programm-Kommunikation zwischen Client- und Server-Programmen
- Mailslots zur Unterstützung der unidirektionalen Programm-Programm-Kommunikation zwischen Client- und Server-Programmen

SPOOL: Drucken in Systemen mit verteilter Verarbeitung

SPOOL von Siemens Nixdorf ist ein Druckservice für verteilte Systeme. SPOOL macht Drucken und Druckverwaltung konsequent einfach. Es werden verschiedene Dienste zur Verfügung gestellt, dazu gehören Verwaltung von Druckaufträgen, leistungsfähige Druckerverwaltung und umfassende Druckfunktionalität.

Die wichtigsten Leistungsmerkmale des SPOOL-Systems sind:

- Möglichkeit des Ausdrucks auf beliebigen angeschlossenen Druckern
- Zentrale Druckerverwaltung
- Auftragssteuerung (Scheduling)
- Lastverteilung, Kontingentierung, Abrechnung (Accounting)
- Arbeitsmöglichkeit auch ohne verteiltes Dateisystem
- Unabhängigkeit von der jeweiligen Netztopologie

SPOOL stellt drei Schnittstellen bereit: eine menügesteuerte Bedienoberfläche, Kommandos auf Shellebene sowie eine Schnittstelle für C-Programme. SPOOL ist objektorientiert zu bedienen, d.h., wenn mit SPOOL ein Auftrag bearbeitet werden soll, wird einer Aktion immer ein Objekt zugeordnet. Wenn man also einer bestehenden Konfiguration einen neuen Drucker hinzufügen will, führt man die Aktion „Gerät hinzufügen" aus, wenn man einen Druckauftrag anstoßen will, führt man die Aktion „Druckauftrag ausführen" aus. SPOOL unterstützt sieben verschiedene Aktionen, die auf 15 verschiedene Objekte angewandt werden können. Dieser objektorientierte Ansatz ermöglicht eine äußerst benutzerfreundliche Bedienung und ist damit ein konsequenter Schritt in Richtung einfacher Handhabung.

Die eigentliche SPOOL-Software besteht aus den fünf Grundkomponenten Client, Server, SPOOL-Systemverwaltung, Datenbankverwaltung sowie Dämon und kann im Netz auf verschiedene Rechner verteilt werden. Die einzelnen Teile kommunizieren über ein eigenes Protokoll, das auf Standardtransportsystemen beruht. SPOOL entspricht den X/Open-Standards für Systemaufrufe und Bibliotheken.

SPOOL unterstützt eine breite Palette von unterschiedlichen Druckern. Aber auch ein neuer Drucker kann durch Definition eines neuen PCL-Objekts (**P**rinter **C**apabilities **L**ist) mit den entsprechenden Druckermerkmalen aufgenommen werden. Dies wird durch folgende Eigenschaften der PCL-Objekte gewährleistet:

- Unterstützung der standardmäßigen Control-Zeichen
- Übersetzung der Optionen eines Druckauftrags
- Datenanalyse
- Dialog mit dem Drucker

SPOOL hat eine Client-Server-Architektur (mehr dazu im nächsten Abschnitt „Verteilte Verarbeitung: DCE als Lösungsmodell" auf Seite 132), arbeitet mit kooperierenden Prozessen (siehe Abschnitt „Kooperative Datenverarbeitung" auf Seite 210) und basiert damit auf modernen Konzepten. Das SPOOL-System hat eine ähnliche Architektur wie Backup (siehe Abschnitt „Backup" auf Seite 104). SPOOL von Siemens Nixdorf ist auch unter dem Namen XPrint auf einer Reihe von Maschinen anderer Hersteller verfügbar; dazu gehören z.B. die SPARC-Systeme von SUN, die ICL DRS-Systeme und die Rechner von Unisys. USL (**U**NIX **S**ystem **L**aboratories) setzt SPOOL in abgewandelter Form als Technologie für das Druckermanagement des Distributed System Management Framework ein.

Verteilte Verarbeitung: DCE als Lösungsmodell

Im Jahr 1990 führte die **O**pen **S**oftware **F**oundation (OSF) eine Ausschreibung durch, an der sich weltweit führende Hersteller beteiligten. Das Ziel dieser Ausschreibung war es, eine umfassende Basis für die Entwicklung und den Betrieb von verteilten Anwendungen zu schaffen: **D**istributed **C**omputing **E**nvironment (DCE). Siemens Nixdorf brachte in diesem Zusammenhang mehrere Komponenten ein. DCE wurde 1993 in kompletter Funktionalität und Qualität als Produkt freigegeben.

OSF sieht DCE als eine Reihe von Diensten und Werkzeugen, die Entwicklung, Einsatz und Pflege von Anwendungen für verteilte Verarbeitung in einem heterogenen Netz unterstützen. DCE ist also eine Art Infrastruktur, die die Entwicklung und den Betrieb von Anwendungen in verteilter Arbeitsumgebung erst ermöglicht.

OSF sieht DCE als eine Vereinigung der Vorteile von offenen Systemen und der verteilten Verarbeitung. Denn DCE ist der erste Versuch, das Konzept der verteilten Verarbeitung in heterogenen Netzen zu unterstützen und dabei herstellerunabhängig von der Hardwarearchitektur und dem Betriebssystem zu bleiben. Mittlerweile hat sich in der Literatur und bei Computerzeitschriften dafür der Begriff „offenes verteiltes System“ eingebürgert. DCE kann dazu benutzt werden, sowohl offene Systeme als auch proprietäre Systeme in einem einzigen, offenen und verteilten System zu integrieren.

Mit DCE wird keine grundlegend neue Funktionalität angeboten, denn alles, was DCE bietet, existiert schon in der ein oder anderen Form. Das Grundlegende und Bahnbrechende an DCE ist die Integration von hochentwickelten Funktionsmerkmalen zu einem neuen Ganzen, das es nun ermöglicht, neue Anwendungen für verteilte Systeme zu entwickeln, ohne daß dabei eine Abhängigkeit von der Verteilungstechnologie eines bestimmten Software-Hauses gegeben ist. Dies garantiert bestmögliche, offene Zusammenarbeit auch unterschiedlicher Systeme.

DCE zielt vor allem auf solche Probleme ab, die bei der Entwicklung von Anwendungen für verteilte Systeme auftreten, wie z.B. eine komplexe Programmierschnittstelle für die Kommunikation zwischen verteilten Komponenten, schlechte Vorhersagbarkeit des Verhaltens und Heterogenität der Programmteile. So blieb bis jetzt die Zahl der Anwendungen für verteilte Systeme beschränkt, da bislang nur die besten Entwickler der Anforderung gerecht werden konnten, den Anwendern die Vorteile einer verteilten Systemumgebung zu erschließen. DCE stellt nun die Grundlagen bereit, damit Entwickler ihre Programme mit den Leistungsmerkmalen versehen können, die ihren Vorstellungen und ihrem Können entsprechen. Hierbei

zeigt DCE seine zwei besonderen Stärken: Dies sind einerseits Remote Procedure Calls, die in der Form den bekannten Local Procedure Calls ähneln und die Zusammenarbeit zwischen Client und Server steuern. Andererseits ist dies die transparente Behandlung der verschiedenen Darstellungsformen von Daten in unterschiedlichen Systemen.

Verteilte Systeme sind nicht notwendigerweise auch offene Systeme. Ein verteiltes System kann z.B. auf ein Netzwerk beschränkt sein, das aus einem homogenen Maschinenpark besteht, also Rechner eines Herstellers, mit gemeinsamer Architektur oder dem gleichen Betriebssystem. Allgemein verbreitet sind heute jedoch Netzwerktypen, hinter denen ein heterogener Maschinenpark steht. Deshalb ist eine der wichtigsten Eigenschaften von DCE die Fähigkeit, auch in einer heterogenen Umgebung ablauffähig zu sein. Genau hier ergänzen sich beide Konzepte, offenes System auf der einen und verteilte Verarbeitung auf der anderen Seite. So ist es durchaus vorstellbar, daß ein Computerhersteller eigens eine Anwendung entwickelt und vermarktet, um den Kunden verteilte Verarbeitung zu ermöglichen, aber nur auf den herstellerspezifischen Systemen. Oder aber der Computerhersteller entschließt sich dazu, einen Schritt weiter zu gehen und auch andere Computerhersteller und deren Systeme mit einzubeziehen, sofern diese die gleiche Architektur oder das gleiche Betriebssystem haben. UNIX System V Release 4 ist ein gutes Beispiel hierfür. DCE von OSF geht aber nochmals einen Schritt weiter, denn DCE ist sowohl unabhängig vom jeweiligen Hersteller als auch vom zugrundeliegenden Betriebssystem. Jeder Computerhersteller kann von OSF eine Lizenz für den Sourcecode von DCE erwerben und DCE auf eigene Rechner portieren. Auf diese Weise entsteht eine Situation, in der die Kunden ein wirkliches offenes System erwerben und Teilnehmer an weltweit verfügbaren, offenen, verteilten Netzwerken werden können.

DCE kommt zwar ursprünglich aus der UNIX-Welt, aber Rechner mit VMS, BS2000, MVS, MS-Windows oder anderen proprietären Betriebssystemen werden in Kürze über DCE-Portierungen verfügen und damit in verteilten Umgebungen arbeiten. DCE baut auf dem TCP/IP-Transportsystem auf, welches auf vielen unterschiedlichen Computersystemen verbreitet ist. DCE schützt die Investitionen der Kunden in ihre Computersysteme, denn durch die hohe Verfügbarkeit wird es möglich, leicht mit anderen Systemen zusammmenzuarbeiten.

Vorteile von DCE

Das Prinzip der verteilten Verarbeitung setzt Unternehmen in die Lage, schon vorhandene Netzwerke, wie LAN oder WAN, optimal auszunutzen. Besonders treffend kommt dies im folgenden Satz zum Ausdruck: „Die eigentlich so unterschiedlichen Konzepte vom offenen System und von der verteilten Verarbeitung hängen doch eng zusammen. Beide Konzepte sind für ein dezentrales Rechnerkonzept in einem modernen Unternehmen unentbehrlich.“ ("The concepts of an open system and a distributed system are separate but strongly related. They are both important in implementing decentralized processing in a contemporary organization.") [Building An Open System, J. Slonim, A. Schonbach, M. Bauer, L. MacRae, and K. Thomas, Van Nostrand Reinhold Company, New York: 1987, S. 3.]

Große, heterogene Netzwerke finden eine immer größere Verbreitung. So verfügt z.B. eine amerikanische Firma über ein firmeneigenes Netzwerk mit über 50 000 Anschlußknoten. Verteilte Verarbeitung bietet die Möglichkeit, innerhalb eines Netzverbundes alle Ressourcen allgemein verfügbar zu machen. Das ist weit mehr, als nur Ressourcen zu teilen, denn aus diesem System wird ein einziger virtueller Computer, dessen Kapazitäten von jedem Knoten innerhalb des Netzes aus erreichbar sind. Das eröffnet ungeahnte Möglichkeiten.

Seit die Leistungsfähigkeit von Server-Rechnern, Workstations und PCs so stark gewachsen ist, existiert das Phänomen, daß diese Leistungsfähigkeit nicht mehr voll ausgenutzt werden kann. Durch das Client-Server-Konzept wird es jedoch möglich, diese brachliegenden Ressourcen wieder zu aktivieren. So ist z.B. vorstellbar, daß die Bearbeitung der Kundenkonten eines Geldinstituts, zuvor auf Großrechnern im nächtlichen Batch-Betrieb durchgeführt, nun auf Workstations verlagert wird: wenn die Angestellten abends nach Hause gehen, arbeiten deren Workstations im Netzverbund die Aufträge ab.

Durch verteilte Verarbeitung können auch spezielle Ressourcen in einem Netzwerk nutzbar gemacht werden, z.B. ein System zur professionellen Bildbearbeitung auf einem Netzrechner, das von jedem beliebigen Netzknoten aus angesprochen werden kann. Es ist naheliegend, daß ein besonders leistungsfähiger Supercomputer extrem rechenintensive Aufgaben für einen Client bearbeitet, oder daß ein leistungsstarker Datenbank-Server einer großen Zahl von Clients, die auf anderen Rechnern liegen, schnellen Zugriff auf eine gemeinsame Datenbank ermöglicht; dabei können die einzelnen Anwender selbstverständlich an ganz unterschiedlichen geographischen Punkten der Erde arbeiten.

Dies ist natürlich nur ein kleiner Ausschnitt der Anwendungen, die in verteilter Umgebung arbeiten. DCE bietet dafür eine Infrastruktur, mit der entsprechende Anwendungen entwickelt werden können. Einige dieser Anwendungen werden Anpassungen von schon bestehenden Anwendungen an die Client-Server-Architektur sein. Daneben wird es völlig neue Anwendungen mit Client-Server-Architektur geben, die dann die verteilte Verarbeitung optimal ausnutzen.

DCE-Komponenten

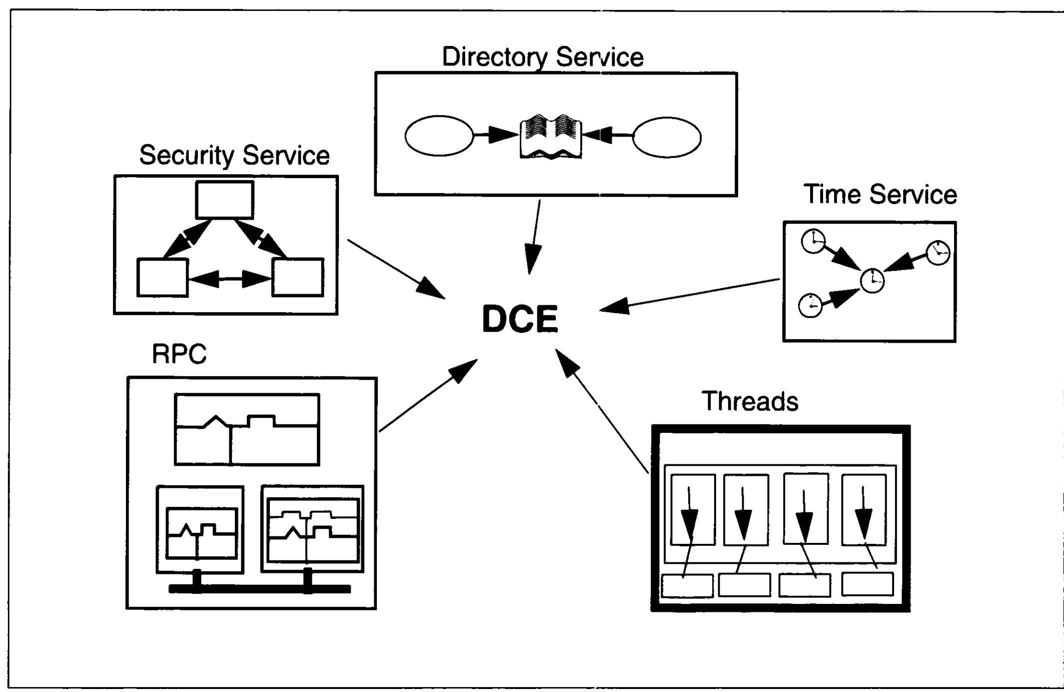


Abbildung 7-1: Die einzelnen DCE-Komponenten bilden eine integrierte Umgebung für Entwicklung und Betrieb von Client-Server-Anwendungen in verteilter Verarbeitung

RPC: Remote Procedure Call

Einen zentralen Bestandteil von DCE bilden die sogenannten **R**emote **P**rocedure **C**alls (RPC), frei zu übersetzen mit: Aufruf ferner Prozeduren. Diese Remote Procedure Calls steuern die Kommunikation zwischen Client- und Server-Komponenten. Andere DCE-Komponenten, z.B. CDS, Security oder Time sind ebenfalls unter Zuhilfenahme von RPC geschrieben. Für Anwendungsprogrammentwickler ist es relativ einfach, Programme mit RPC zu schreiben, denn sie stellen nur eine Erweiterung der schon geläufigen Prozeduraufrufe für eine Netzumgebung dar.

Eine aufrufende Prozedur und die aufgerufene, lokale Prozedur werden im gleichen Adreßraum (Prozeß) ausgeführt. Eine ferne Prozedur hat einen anderen Adreßraum (Prozeß) zur Verfügung, da sie auf einem anderen Rechner abläuft. Die RPCs ersparen es dem Anwendungsprogrammentwickler, sich um diese Dinge zu kümmern. Die bereitgestellten DCE-Dienste meistern alle schwierigen Aufgaben: sie finden den geeigneten fernen Rechner, bauen die Verbindungen auf, kontrollieren die Kommunikation zwischen Client und Server und verarbeiten auch unterschiedliche Datenformate der einzelnen Systeme.

Bei der Entwicklung einer DCE-Anwendung ist darauf zu achten, daß streng zwischen der Server- und der Client-Komponente des Programms unterschieden wird. Deshalb müssen Schnittstellen zwischen Server und Client festgelegt werden, die in einer C-ähnlichen Sprache, der **I**nterface **D**efinition **L**anguage (IDL), geschrieben sind.

Einer der wichtigsten Vorteile der RPCs in DCE ist die Verwendbarkeit der zugrundeliegenden Semantik auch in unterschiedlichen Transportsystemen. So werden zur Zeit die verbindungslosen Dienste über UDP/IP und die verbindungsorientierten Dienste über TCP/IP unterstützt. Ein Programmentwickler braucht lediglich das gewünschte Transportsystem auszuwählen.

Durch die zugrundeliegenden Threads (deutsch: Stränge; Unterprozesse) können mit RPC innerhalb eines Prozesses mehrere Aktivitäten parallel ablaufen. Ein Client kann mehrere parallele RPCs aussenden, die Server führen diese Aufrufe auch parallel aus. Diese Fähigkeit bringt große Leistungsvorteile bei der verteilten Verarbeitung, denn dadurch wird eine Anwendung hierarchisch in Aufgaben aufgeteilt, die dann wiederum auf verschiedenen Prozessoren parallel bearbeitet werden können.

Die RPCs sind auch eng in die Security- und Directory-Dienste von DCE eingebunden, um sichere RPCs auf transparente Weise zu unterstützen. Die gewünschte Sicherheitsstufe wird dabei in der Schnittstellenbeschreibung festgelegt. Der Directory-Dienst macht es möglich, daß Clients automatisch den passenden Server finden können.

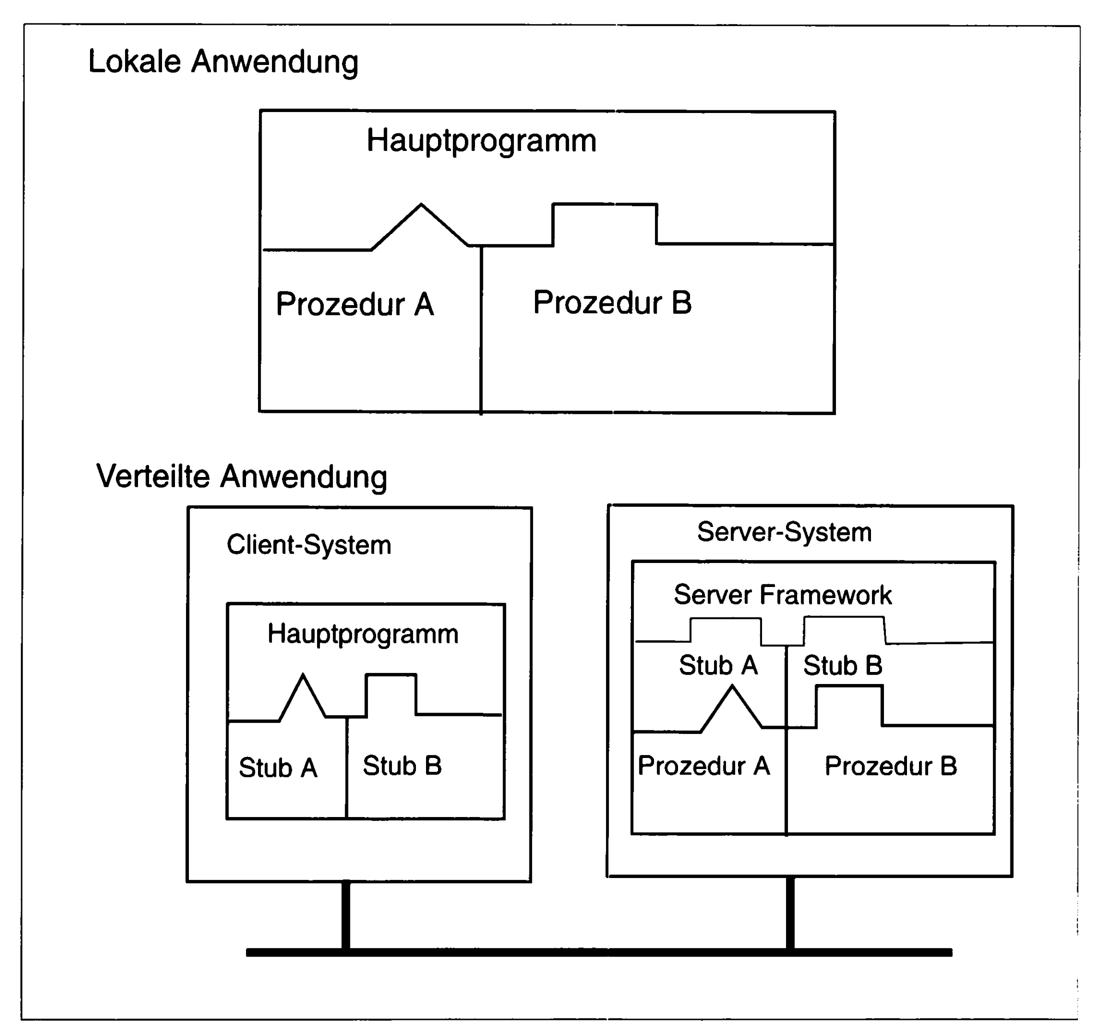


Abbildung 7-2: RPCs erweitern Mechanismen, wie sie von Local Procedure Calls her bekannt sind, zur Unterstützung der Client-Server-Kommunikation.

Das ist natürlich nur ein Ausschnitt, denn ein Remote Procedure Call erfordert einen höheren Programmieraufwand als ein Local Procedure Call, weil das Hauptprogramm und die Prozedur nicht mehr Teil desselben Prozesses sind. Sowohl der Client-Prozeß mit dem Hauptprogramm als auch der Server-Prozeß mit der Prozedur müssen verläßlich zusammenarbeiten und miteinander kommunizieren. Der Grad der Sicherheit ist dabei abhängig von der Art der Netzverbindung, die z.B. vor Hackern sicher bleiben soll. Beim eigentlichen Datenaustausch zwischen Client und Server kann man noch einen höheren Grad der Vertraulichkeit einbauen, z.B. durch Verschlüsselung der Daten. Dies alles zeigt, warum Remote Procedure Calls allein noch kein DCE ausmachen.

Threads

In einem System mit verteilter Verarbeitung benötigt man Threads zur Kontrolle der Vorgänge innerhalb eines Prozesses. Normalerweise braucht man für einen DCE-Anwendungsserver pro aktiver Anfrage eines Clients auch jeweils einen Thread. Damit wird die Parallelverarbeitung verschiedener Aufgaben innerhalb eines Prozesses möglich. In UNIX erreicht man normalerweise eine ähnliche Funktionalität, indem man einen Prozeß vervielfältigt. Es ist allerdings weit weniger aufwendig, mit Threads zu arbeiten, als einen Prozeß zu vervielfältigen, zumal sich Threads auch wesentlich besser gemeinsame Daten teilen können. Normalerweise arbeitet nur das Betriebssystem mit Threads, DCE-Threads sind jedoch betriebssystemunabhängig, alle weiteren DCE-Komponenten bedienen sich dieser Threads.

Das Konzept der DCE-Threads basiert auf dem POSIX-Entwurf 1003.4a „Threads Extension for Portable Operating Systems“ (Threadserweiterung für POSIX). Dieser Entwurf bildet auch die Grundlage für die Verwendung des Threads-Konzepts in UNIX, so daß schon existierende Anwendungen, die auf dem Konzept der DCE-Threads beruhen, angepaßt und unter einem UNIX mit Threads-Funktionalität rasch eingesetzt werden können.

Ein Serverprozeß wird normalerweise nicht nur von einem einzigen Client angesprochen. Man könnte theoretisch alle RPCs streng der Reihe nach abarbeiten lassen. Wenn nun aber die Abarbeitung eines RPCs mit Wartezeiten verbunden ist, dann werden währenddessen völlig unnötigerweise alle anderen Anforderungen auch zum Warten gezwungen. Ein Server wird deshalb normalerweise versuchen, eine Reihe von RPCs parallel abzuarbeiten, wozu DCE die Threads-Komponente bereitstellt. Threads sind sozusagen kleine Prozesse innerhalb eines Prozesses. Mit Threads können mehrere Aktivitäten innerhalb eines Prozesses parallel ablaufen.

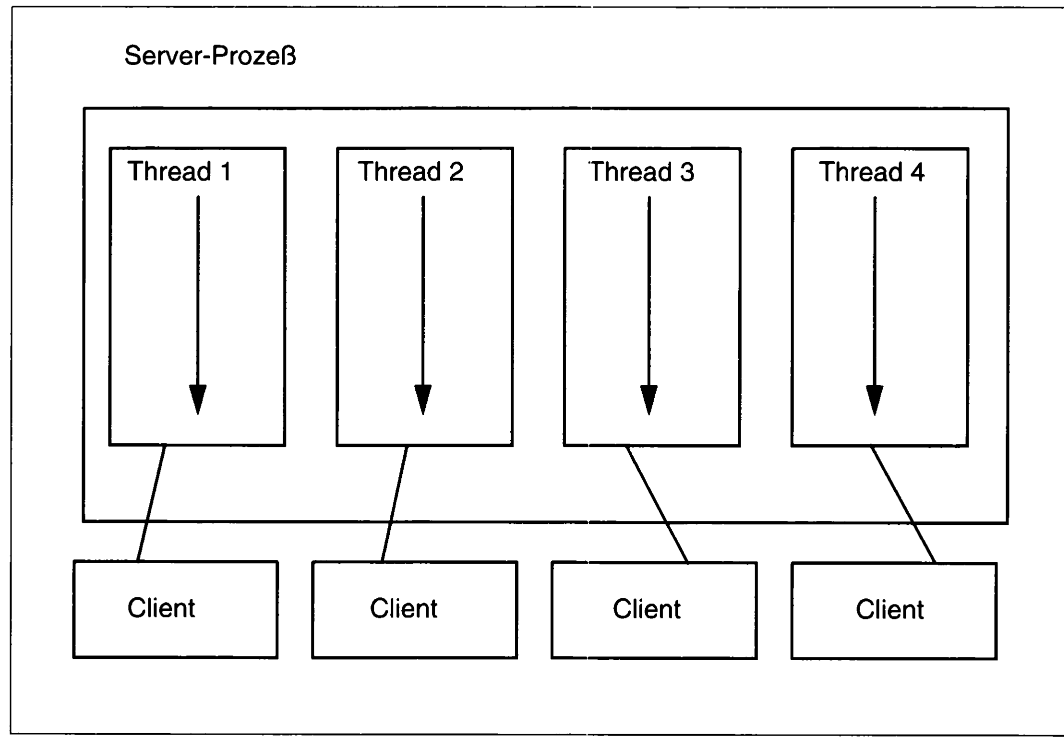


Abbildung 7-3: Server benutzen Threads zur simultanen Steuerung der Beziehungen zu den einzelnen Clients.

Auch Clients können auf Threads zugreifen, z.B. wenn ein Client schnell bestimmte Informationen von einigen Servern erhalten möchte. Ein konkretes Fallbeispiel: ein zentraler Client, Anwendung für eine Einzelhandelskette, fragt Daten von den Servern, den einzelnen Läden der Kette, ab und erstellt aus diesen Daten dann einen Ergebnisbericht. Mit Hilfe der Threads kann ein Client weiterhin neue Anfragen abschicken, gleichzeitig auf die Antworten warten und daneben die eingehenden Daten ordnungsgemäß bearbeiten.

Der Einsatz dieser Threads erfordert einen hohen Aufwand bei der Programmierung von Serverkomponenten, da es sich hier um eine Form der Parallelprogrammierung handelt. Es ist besonders darauf zu achten, daß Daten, die von mehreren Threads benutzt werden, auch besonders geschützt werden. Die Threads-Komponente enthält jedoch alle hierfür nötigen Sperr- und Synchronisationsmechanismen.

CDS: Cell Directory Service

Mit DCE verfolgt man das Konzept, viele Computersysteme zu einem einzigen virtuellen System zu vereinigen; deshalb ist es auch notwendig, bestimmte Namenskonventionen festzulegen, damit einzelne Objekte dieses Systems eindeutig identifizierbar bleiben. Wenn es nur einige wenige Dateiverzeichnisse gibt, z.B. */usr* oder */bin*, ist dies natürlich nicht problematisch. Wenn die Zahl der Dateiverzeichnisse oder anderer Ressourcen, wie z.B. Drucker, in die Hunderte oder gar Tausende geht, muß ein neues Konzept der Namensgebung eingesetzt werden.

DCE bietet für diesen Bereich den **C**ell **D**irectory **S**ervice (CDS). Eine Zelle ist eine beliebige administrative Einheit aus mehreren DCE-Rechnern. Häufig wird eine Zelle aus einem lokalen Netz (LAN) gebildet, was aber keine notwendige Voraussetzung für DCE ist. Eine Zelle kann auch aus dem Zusammenschluß mehrerer LANs oder aus dem Zusammenschluß mehrerer LANs mit fernen Rechnern gebildet werden. Ebenso kann eine Zelle aus einer einzigen Maschine bestehen, aus einem Dutzend oder aus Tausenden. Das Zellenkonzept wurde in den CDS eingeführt, um Einheiten, wie die eben dargestellten, benennen zu können. Eine Minimalkonfiguration einer Zelle besteht aus Authentication-Server, Time-Server und CDS-Server.

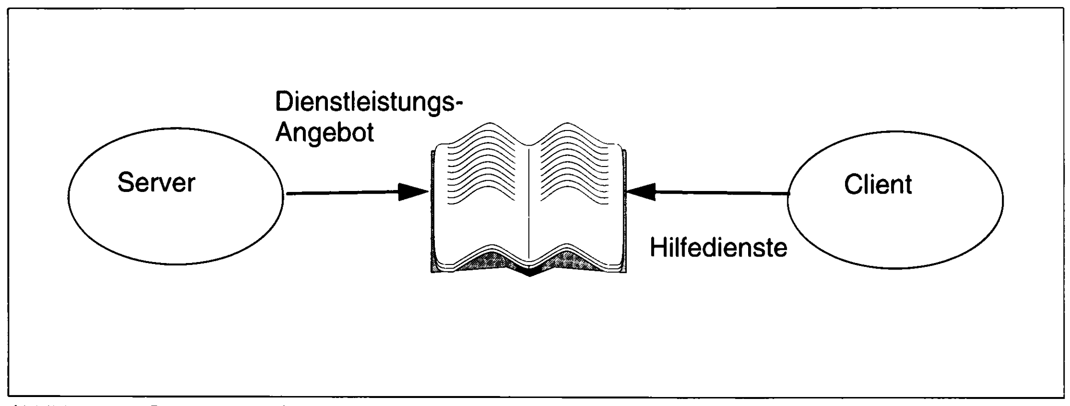


Abbildung 7-4: Der Directory Service ist mit den gelben Seiten eines Telefonbuchs vergleichbar. Server bieten ihre Dienste an, Clients nehmen diese in Anspruch.

Wenn DCE in großen, geographisch getrennten, verteilen Netzverbünden arbeitet, benötigt man zur Organisation und Strukturierung dieser riesigen Arbeitsumgebung sogenannte Zellen. Die Zellen haben natürliche Grenzen, innerhalb dieser Strukturen können Sicherheitsabfragen durchgeführt, Namen vergeben und Dateisysteme verwaltet werden. Die Zellengrenzen schränken nicht die Kommunikationsfähigkeit der einzelnen Rechner untereinander ein, die DCE-Architektur kann aber für bestimmte Operationen innerhalb dieser Grenzen optimiert werden.

Der Directory-Service unterstützt das Auffinden der einzelnen Dienste im DCE-Netz. Sobald ein Server seinen Betrieb aufnimmt, meldet er sich beim Directory-Service an. Auf diese Weise ist der Directory-Service immer auf dem neusten Stand und kann seiner Aufgabe als Wegweiser im DCE-Netz nachkommen. Ein Client, der einen Dienst benötigt, macht seinerseits eine entsprechende Anfrage an den Directory-Service. Ein und derselbe Dienst kann mehrfach im Netz angeboten werden, ein Client kann dann entweder zufällig einen passenden Dienst in Anspruch nehmen oder aber einen entsprechenden Dienst nach bestimmten Kriterien auswählen.

Weitere wichtige Eigenschaften des Directory-Service sind hohe Verfügbarkeit durch parallele Datenhaltung, gute Leistung durch Schreiben der Daten in Puffer auf den einzelnen Knotensystemen, Eignung sowohl für kleine als auch für große Datenmengen durch hierarchische Strukturierung der Verzeichnisinformationen.

Synchronisation der Systemzeiten: Time-Service

DCE bietet mit dem **D**istributed **T**ime **S**ervice (DTS) einen Dienst zur Zeitsynchronisation an. DTS verfügt über eine Schnittstelle zu einem externen Zeitgeber (z.B. das Signal eines Zeitzeichensenders). Synchrone Systemzeiten sind für die funktionierende Zusammenarbeit unbedingt notwendig, weil nur dann die Zeitmarken, die auf verschiedenen Rechnern erzeugt wurden, sinnvoll verglichen werden können und die Zeitbestimmung für bestimmte Ereignisse über Rechnergrenzen hinweg exakt ist. Die meisten verteilten Algorithmen, die z.B. von CDS, dem Sicherheitsdienst und DFS benutzt werden, funktionieren nur dann einwandfrei, wenn die Systemzeiten synchron laufen.

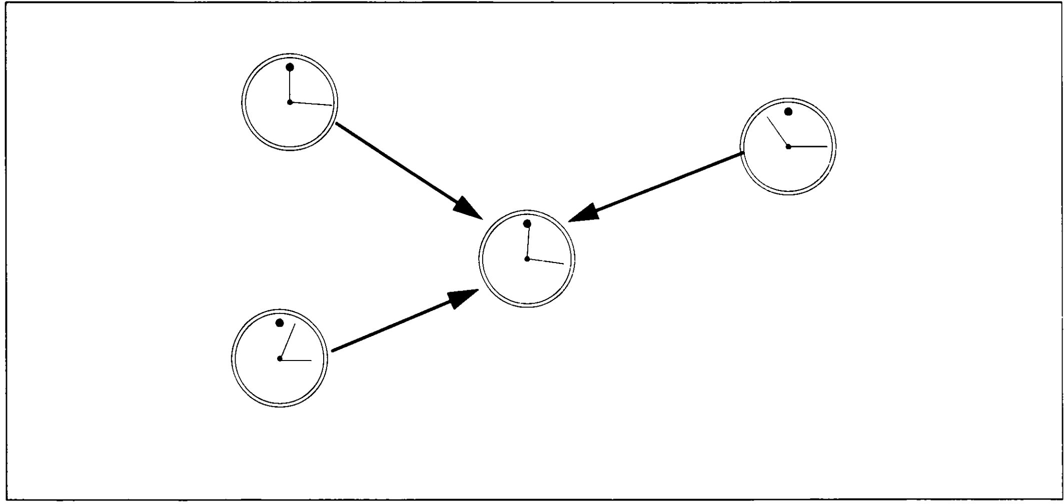


Abbildung 7-5: Der Time-Service sorgt für die Synchronisierung der einzelnen Systemzeiten.

Sicherheit: Security-Service

DCE bietet mit dem Security-Service Dienste an, die es ermöglichen, in verteilten Systemen auch über unsichere Verbindungen miteinander zu kommunizieren und dabei über ein hohes Maß an Sicherheit beim Zugriff auf Ressourcen zu verfügen. Der DCE-Security-Service überprüft drei wichtige Aspekte:

- Identität
- Autorisierung
- Integrität der Kommunikation

Über die Dienste des Directory-Service können sich zwei Partner (Client und Server) ausfindig machen. Es stellt sich nun für die Partner die Frage, ob sein Gegenüber wirklich der ist, für den er sich ausgibt. Der Server möchte natürlich wissen, wer seine Dienste in Anspruch nehmen möchte, damit später die erbrachten Leistungen abgerechnet werden können. Der Security-Service bietet dazu den Authentisierungs-Service an, vergleichbar einem verteilten „login". Dieser Vorgang wird bisher mit Paßwörtern durchgeführt, aber an einer Integration von neuen Sicherheitstechnologien (z.B. intelligenten Chipkarten) wird derzeit gearbeitet. Nach der Abfrage kennt der Security-Service zuverlässig die Identität von Client und Server und tritt dann als Vermittler zwischen den Partnern auf.

Ein weiterer Aspekt ist die Kontrolle des Zugangs und der Zugriffsrechte: wer darf was im DCE-Netz. Der Security-Server bietet durch Verwaltung der Zugriffsdaten auch hierfür geeignete Unterstützung: Über **A**ccess **C**ontrol **L**ists (ACL) lassen sich, falls erforderlich, für jedes einzelne Objekt und jeden einzelnen Benutzer die Zugriffsrechte regeln.

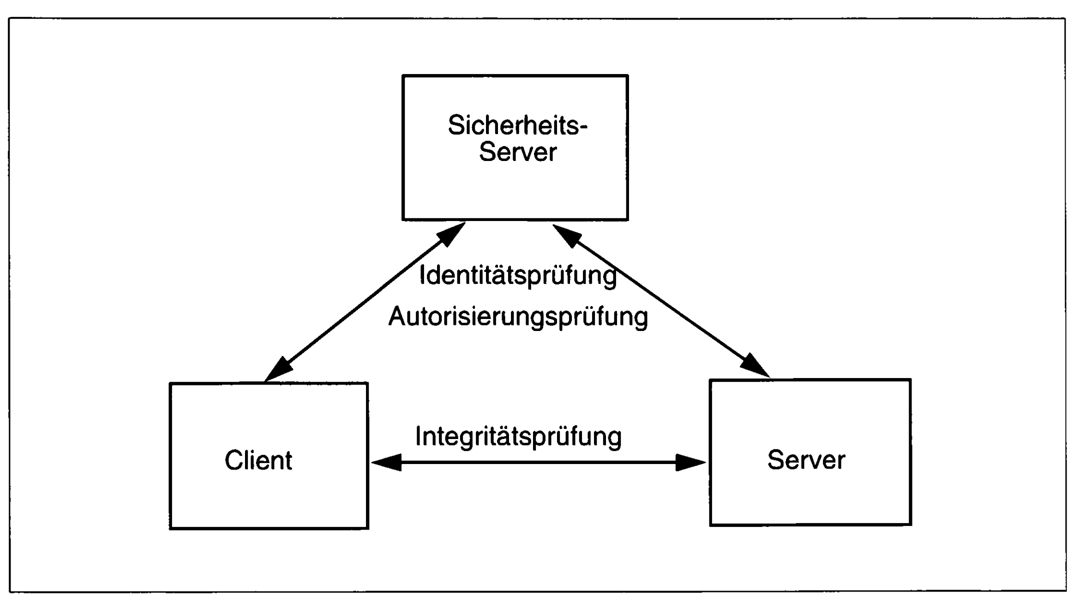


Abbildung 7-6: Der Security Service überprüft Identität, Autorisierung und Integrität.

Beim eigentlichen Datenaustausch werden verschiedene Ebenen der Integrität garantiert (z.B. Schutz vor verfälschten Nachrichten) sowie Vertraulichkeit (z.B. Datenverschlüsselung). Naturgemäß verursacht ein höheres Sicherheitsniveau auch eine höhere Inanspruchnahme des Systems. Deshalb kann der Progammentwickler je nach Notwendigkeit das geeignete Sicherheitsniveau angeben. Das RPC-Laufzeitsystem kümmert sich dann in Zusammenarbeit mit dem Security-Service um die effektive Ver- und Entschlüsselung der Daten. Die Benutzung eines gesicherten RPCs unterscheidet sich also nicht von einem beliebigen anderen RPC.

GDS: Global Directory Service

Der **G**lobal **D**irectory **S**ervice (GDS), eine Entwicklung von Siemens Nixdorf, erlaubt die Verbindung zwischen den verschiedenen lokalen Zellen des globalen DCE-Netzes. GDS ist ein internationaler Verzeichnis-Dienst gemäß dem OSI bzw. X.500-Standard (X.500 ist die CCITT-Bezeichnung desselben Standards). Er wird von Anwendungen und Benutzern zum Speichern von Namen verwendet und stellt weitaus bessere Recherchemöglichkeiten als CDS bereit.

DFS: Distributed File System

Was die Zugriffsmöglichkeit auf ferne Dateien anbelangt, übertrifft das **D**istributed **F**ile **S**ystem (DFS) das konventionelle NFS-Dateisystem bei weitem. DFS bietet eine einheitliche, systemunabhängige und hierarchische Namensgebung. Die verwendete Semantik entspricht voll dem POSIX-Standard, einschließlich der netzweiten Sperrmechanismen. Teile der hierarchisch strukturierten Dateiverzeichnisse können mehrfach gehalten werden, um eine hohe Verfügbarkeit und eine gleichmäßige Auslastung zu erreichen. Um die Zugriffszeit auf Dateien zu verringern, werden Teile der Dateien in Puffern gehalten. DFS ist in besonderer Weise für die Verwaltung von fernen Systemen ausgelegt und reduziert damit den Verwaltungsaufwand für Dateisysteme in großen Netzwerken. DFS ist voll kompatibel zu NFS und bietet einfache Migrationswege für alle NFS-Anwendungen.

DFS baut innerhalb einer DCE-Zelle einen Dateibaum auf, bestehend wiederum aus Teilbäumen, die die einzelnen Systeme über DFS exportieren. Im Unterschied zu NFS sieht dieser Baum für alle beteiligten Systeme gleich aus. DFS implementiert auf seinen Dateien die volle Semantik eines lokalen Dateisystems, insbesondere gelten die gleichen Garantien für die Reihenfolge der Datenzugriffe und es gibt wirkungsvolle Sperrmechanismen.

Das verteilte Dateisystem (DFS) ist einerseits eine Komponente des OSF/DCE, es kann aber auch als Anwendung auf der Basis der anderen Dienste betrachtet werden.

DFS ist voll kompatibel zum bestehenden UNIX-Dateisystem, bietet aber noch ein eigenes Dateisystem. Die Vorteile des neuen, lokalen Dateisystems sind geringe Ausfallzeiten nach einem Systemabsturz, die Möglichkeit der dynamischen Teilbaumverlagerung und Tabellen zur Zugangskontrolle. DFS und sein fortschrittliches Dateisystem bieten zusammen die interessante Möglichkeit, Teilbäume physikalisch zu verlagern, ohne daß sich für den Benutzer der Zugriffspfad ändert, ja sogar ohne daß die Benutzung des Teilbaums unterbrochen werden müßte. Flexibilität in der Konfigurierung, hohe Verfügbarkeit durch kurze Ausfallzeiten und ein Online-Backup-System sind Argumente, die DFS zu dem verteilten Dateisystem der Zukunft machen wird.

Die DFS-Komponente wird zu einem späteren Zeitpunkt als Produkt zur Verfügung stehen, als die Grunddienste von DCE. Das ist zum einem in der Reihenfolge der Entwicklung begründet, denn DFS baut auf allen Diensten der Grundversorgung auf, aber zum anderen auch in der Komplexität und Innovationskraft dieser Technologie, sowie schließlich in den höheren Qualitätsanforderungen, die an einen so grundlegenden Dienst wie ein Dateisystem gestellt werden.

Verfügbarkeit von DCE

DCE wird von vielen Herstellern gemeinsam zur Verfügung gestellt. Auf den CeBIT-Messen der Jahre 1991 und 1992 führte Siemens Nixdorf in Zusammenarbeit mit IBM, HP, Bull, Stratus und OSF eine Demonstration von DCE vor. Das Engagement dieser und anderer führender internationaler Anbieter stellt sicher, daß DCE sich als De-facto-Industriestandard durchsetzen wird. So verwendet derzeit X/Open viel Mühe darauf, die Spezifikation von DCE in das Common Applications Environment (X/Open CAE) zu übernehmen. Die ersten DCE-Dienste, die in das X/Open CAE aufgenommen werden, sind RPC und der Time-Service. Obwohl DCE ursprünglich in der UNIX-Welt entstand, wird es nicht darauf beschränkt bleiben. Die Basisdienste von DCE lassen sich auch auf anderen Betriebssystemen bereitstellen. Es gibt bereits entsprechende Ankündigungen sowohl aus der PC-Welt als auch aus dem Mainframe-Bereich.

DCE-Produkte von Siemens Nixdorf

Als Hersteller der GDS-Komponenten war Siemens Nixdorf von Anfang an mit der Entwicklung von DCE befaßt. Siemens Nixdorf wird auch in Zukunft dafür sorgen, daß DCE auf unterschiedlichen Systemfamilien, wie SINIX, Windows oder BS2000, verfügbar ist.

DCE (SINIX)

DCE (SINIX) ist eine vollständige Implementierung von DCE in das SINIX-Betriebssystem. Das Programm wurde von einem internationalen Entwicklerteam aus Deutschland und den USA bei Siemens Nixdorf entwickelt. OSF hat die Portierung von DCE auf SINIX (UNIX SVR4) als Referenzportierung für diese Betriebssystembasis ausgewählt. Infolgedessen kam ein Kooperationsabkommen zwischen USL (UNIX System Laboratories), Vertreiber von UNIX SVR4, und Siemens Nixdorf zustande, wonach Siemens Nixdorf im Rahmen dieses Vertrags die Portierung von DCE an USL zur Weitergabe an alle SVR4-Lizenznehmer liefert. Damit ist DCE von Siemens Nixdorf das standardmäßige DCE-Produkt auf allen SVR4-UNIX-Systemen.

DCE (SINIX) ist in Form verschiedener Produkte Siemens Nixdorf verfügbar:

Das Modul **DCE Executive** enthält das RPC-Laufzeitsystem, Threads und die Directory- und Security-Clients.

Das Modul **DCE Developer** enthält die Werkzeuge, die zur Entwicklung von DCE-Anwendungen gebraucht werden.

Das Modul **DCE Starter** enthält alle Werkzeuge, die zur Erzeugung einer DCE-Zelle benötigt werden. Es stellt darüber hinaus die Komponenten für die Entwicklung und die Ablaufumgebung von DCE-Anwendungen zur Verfügung.

Der **DCE Cell Directory Service** ist der CDS-Server. Pro DCE-Zelle wird davon je ein Server benötigt.

Der **DCE Security Service** ist der Sicherheits-Server. Darin eingeschlossen sind Verwaltung, Glaubwürdigkeitsprüfung und Ermächtigungsprüfung.

Der **DCE Crypto Service** erweitert den Leistungsumfang von DCE-Executive. Im Zusammenwirken mit dem Security-Server können Daten verschlüsselt werden.

DCE (WIN)

DCE (WIN) läuft unter MS-Windows auf PCs und stellt die Verbindung zwischen PCs und DCE-Netzwerken her. DCE (WIN) ist eine MS-Windows 3.x Implementierung des OSF/DCE. Dadurch kann auch ein PC als DCE-Client arbeiten (trusted peer). DCE (WIN) wird als **D**ynamic **L**ink **L**ibrary (DLL) zur Verfügung gestellt, weshalb Windows-Anwendungen vollen Zugriff auf die DCE-Anwendungsprogrammierschnittstelle (API) erhalten.

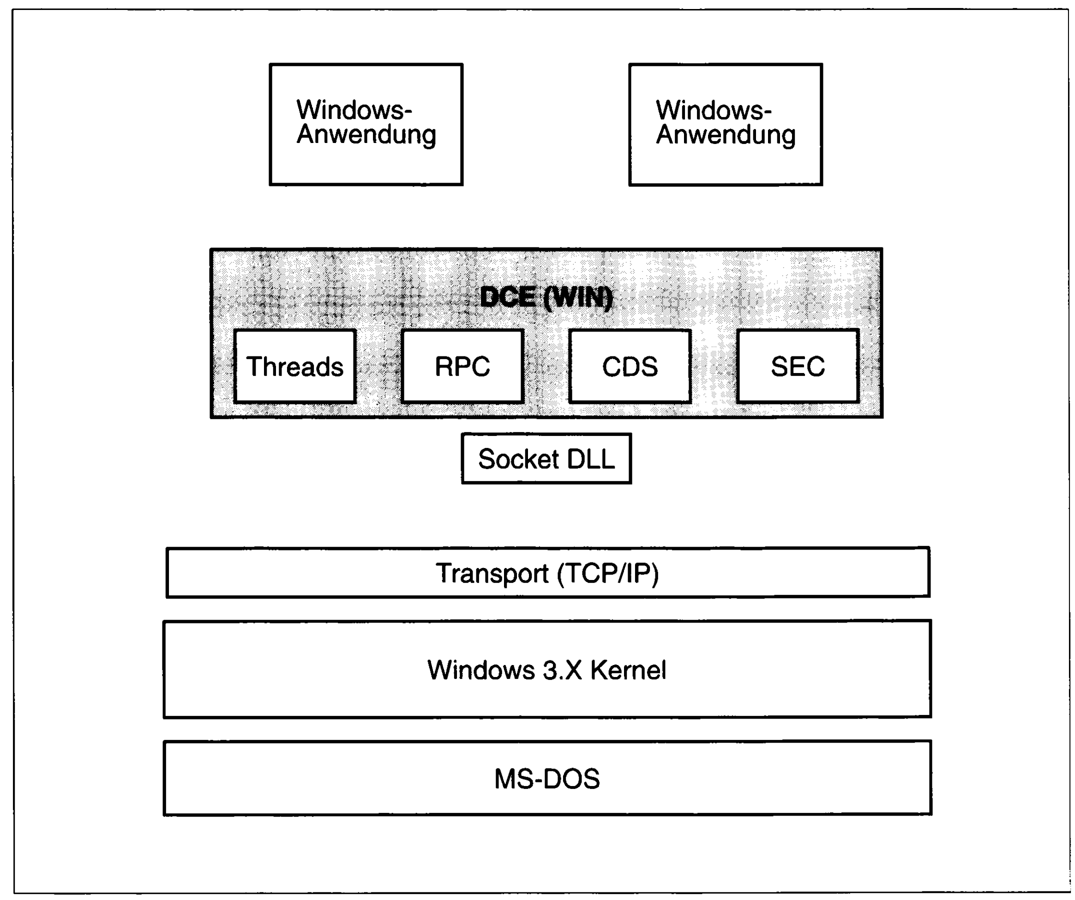


Abbildung 7-7: DCE (WIN) schafft die Voraussetzungen, um PCs als Clients in Arbeitsumgebungen mit verteilter Verarbeitung einzubinden.

Die Implementierung von DCE als DLL bringt als großen Vorteil eine Verringerung des Speicherplatzbedarfs, denn DLL-Routinen werden erst dann in den Arbeitsspeicher geladen, wenn sie von einer Anwendung angefordert werden und können dann auch von anderen Anwendungen gemeinsam genutzt werden. Darüber hinaus benötigen DLLs innerhalb der einzelnen Anwendung keinen Adreßraum. Das DLL für DCE (WIN) stellt also DCE-Dienste zur Verfügung, ohne anderen PC-Anwendungen wertvollen Speicherplatz zu nehmen. DCE (WIN) bildet die Verbindung zwischen DCE und Standardanwendungen unter Windows. Damit können Standard-Windows-Anwendungen so konzipiert werden, daß PCs gleichwertige Funktionalität innerhalb eines DCE-Netzverbunds erhalten. Mit Hilfe der von Windows zur Verfügung gestellten Dienste können Daten direkt in Windows-Anwendungen abgebildet werden. Auf diese Weise können gängige Windows-Anwendungen für die Anwender einfach zu einer bequemen Schnittstelle in einem DCE-Netzwerk gemacht werden.

DCE (WIN) umfaßt verschiedene DCE-Komponenten. Dazu gehören der Secure-Core-DCE-Client mit Threads-Diensten, Client/Server-Funktionalität des RPC, Client-Funktionalität des Cell Directory Service, Security Service und Time Service. Es stehen darüber hinaus zwei Versionen mit jeweils unterschiedlichen Sicherheitsstufen zur Verfügung: die Vollversion mit hoher Sicherheitsstufe verfügt über Verschlüsselungsmechanismen auf Anwendungsebene, die eingeschränkte Version unterstützt das DCE-Verschlüsselungsprotokoll. DCE (WIN) ist als Laufzeitversion und als Entwicklungswerkzeug (Application Developer´s Kit) verfügbar.

Das **A**pplication **D**eveloper´s **K**it (ADK) bietet Entwicklern die breite Palette der zu Windows kompatiblen Ressourcen und Werkzeuge, mit denen die volle Leistung von Windows-Anwendungen für DCE gewährleistet werden kann. Das Laufzeitmodul enthält die Laufzeitbibliotheken, die Installationsprogramme und die Benutzerdokumentation. Es enthält auch die Programmteile, die auf einem PC installiert werden müssen, um die mit dem ADK entwickelten Programme ablauffähig zu machen. Darüber hinaus bietet es weitere Programmkomponenten, wie z.B. ein DCE-Login-Programm, Laufzeitserver (RPC Dämon, CDS Clerk) und diverse Dienstprogramme (z.B. RPC und CDS-Kontrollprogramm, ACL Edit). Mit dem Laufzeitmodul kann eine eingeschränkte Version von DCE (WIN) erzeugt werden, die Verschlüsselungsoptionen auf DCE-Systemebene bietet. Soll eine Vollversion erzeugt werden, ist dazu das DCE (WIN) Crypt-Modul nötig.

DCE (BS2000)

DCE (BS2000) wird es ermöglichen, Anwendungsserver auf BS2000-Systemen zu installieren. BS2000-Systeme werden somit für Clients auf PCs und auf SINIX-Systemen nutzbar. Wegen des hohen Integrationsaufwands rechnen wir mit der Verfügbarkeit im zweiten Quartal 1994.

150

8 SINIX-Anwendungen im kommerziellen Großeinsatz

Überblick

Typische Dialoganwendungen wie Buchungs-, Auskunfts-, Lagerhaltungs-, Bestell- oder Produktionssteuerungssysteme haben die Eigenschaft gemeinsam, daß an einer großen Zahl von Terminals oder Workstations voneinander unabhängig Aufträge erteilt werden und mit aktuellen Daten aus zentralen Datenhaltungssystemen, in der Regel Datenbanken, verarbeitet werden sollen. Die von diesen Anwendungen gemeinsam benötigten Leistungen und das Überwachen und Steuern der Ressourcen, wie z.B. Datenendgeräte mit Datenstationen und Druckern, gemeinsame Speicherbereiche im Verarbeitungsrechner, Prozesse, Programme usw., werden von einem **T**ransaction-**P**rocessing-Monitor (TP-Monitor) übernommen, unter dessen Steuerung die Anwendungen ablaufen.

Ein TP-Monitor wird auch als Datenkommunikationssystem oder DC-System bezeichnet. Zusammen mit Datenbanksystemen wie z.B. INFORMIX, ORACLE, oder UDS, ergibt ein TP-Monitor ein DB/DC-System. Solche Kombinationen werden in der Fachliteratur auch als OLTP-Systeme (**On** **L**ine **T**ransaction **P**rocessing) bezeichnet. Die Grundstruktur eines OLTP-Systems ist in Abbildung 8-1 dargestellt.

Unter einer Transaktion versteht man eine Folge von logisch verknüpften Arbeitsschritten, die entweder vollständig oder überhaupt nicht ausgeführt wird. Ein einfaches Beispiel ist die Überweisung eines Geldbetrags von einem Konto auf ein anderes. Indem die Abbuchung vom einen Konto und die Überweisung auf ein anderes Konto zu einer einzigen Transaktion zusammengefaßt werden, verhindert man, daß nur ein Arbeitsschritt vollendet und ein anderer ausgelassen wird. Denn sonst könnte es geschehen, daß zwar Abbuchungen von einem Konto vorgenommen werden, die dazugehörigen Überweisungen auf das andere Konto jedoch un-

terbleiben. Tritt der Fall ein, daß eine Transaktion nicht erfolgreich beendet werden konnte, werden automatisch alle an dieser Transaktion beteiligten Daten in den ursprünglichen Zustand zurückversetzt.

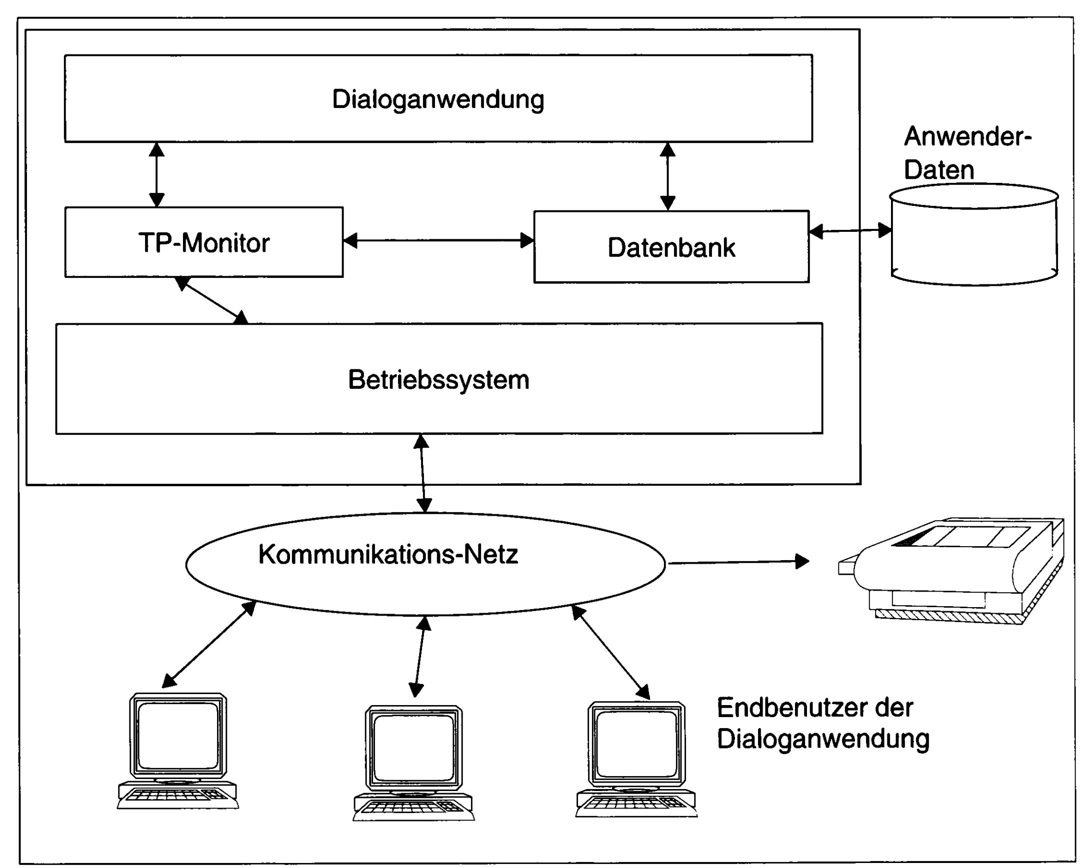


Abbildung 8-1: Grundstruktur eines OLTP-Systems.

OLTP mit UTM (SINIX)

OLTP auf SINIX-Systemen wird schon seit Jahren erfolgreich durchgeführt und dabei in steigendem Umfang im unterbrechungsfreien 24-Stunden-Betrieb. Dadurch wird jeweils einer großen Zahl von Anwendern die gleichzeitige Benutzung ermöglicht. Siemens Nixdorf kann diesbezüglich auf einen reichhaltigen Erfahrungsschatz aus der Mainframe-Welt zurückgreifen. Die Produkte, die für die Großrechner im kommerziellen Bereich entwickelt wurden, sichern einen enormen Wissensvorsprung bei der Anpassung von SINIX an Programme für den kommerziellen Großeinsatz.

Ein Beispiel dafür ist die erfolgreiche Portierung von UTM, dem Transaktionsmonitor von Siemens Nixdorf aus der Mainframe-Welt auf UNIX-basierte Systeme im Jahr 1988. Mit mehr als 700 Installationen (Stand: Sommer 1992), ist UTM (SINIX) der am häufigsten eingesetzte Transaktionsmonitor auf UNIX-Systemen in Europa.

OLTP-Systeme benötigen konsistente Daten und müssen gegenüber allen äußeren Störungen unempfindlich sein. Diese grundlegenden Anforderungen an die Transaktionsverarbeitung werden auf SINIX-Systemen mit einem Datenbanksystem (z.B. INFORMIX) und dem Transaktionsmonitor UTM erfüllt. Sie kommunizieren untereinander über die von X/Open als Standard festgelegte XA-Schnittstelle.

SINIX wird häufig zusammen mit anderen Systemen im Netzverbund eingesetzt, so z.B. mit anderen SINIX-Systemen, mit Systemen aus der BS2000-Welt, der MS-DOS-Welt und anderen Systemen. Voraussetzung für eine umfassende Lösung für den Anwender ist, daß OLTP-Anwendungen und Daten im Netz verteilt gehalten werden können. Dabei ist die räumliche Verteilung für den Anwender nicht sichtbar, genausowenig wie erkennbar wird, an welchem Ort und von welchem System eine Transaktion durchgeführt wird. Das Netzwerk erscheint für den Benutzer als eine logische Einheit. Obwohl es sich dabei um technisch höchst komplexe und räumlich verteilte Systeme handelt, garantieren UTM und INFORMIX stets den reibungslosen Ablauf der Transaktionen sowie konsistente Datenhaltung.

Siemens Nixdorf ist mittlerweile ein führender Anbieter von verteilten Transaktionssystemen und Datenbanksystemen. Die vielfältigen Möglichkeiten der Konfiguration verteilter Transaktions- und Datenbankverarbeitung mit UTM und INFORMIX stellen allen Kunden eine jeder Situation angepaßte Lösung zur Verfügung.

Anforderungen an einen Transaktionsmonitor

Ein Transaktionsmonitor muß den folgenden Anforderungen genügen:

- der Benutzerdialog wird von der jeweiligen Anwendung gesteuert
- sehr viele Benutzer müssen gleichzeitig daran arbeiten können
- die Daten einer derartigen Anwendung müssen immer konsistent sein
- die Anwendung muß unempfindlich gegenüber Störungen von außen sein; dies bedeutet im einzelnen,
  - daß ein Verbindungsverlust keinen Einfluß auf die Gesamtanwendung hat; ein vom Ausfall betroffener Benutzer kann seine Tätigkeit definiert fortsetzen.
  - daß der Ausfall einer Gesamtanwendung nur zu einer temporären Betriebsunterbrechung führt, bis ein automatischer Wiederanlauf durchgeführt werden kann.
  - daß die Wirkung von Fehlern in Anwendungsprogrammen lokal bleibt.
  - daß eine Anwendung vor unberechtigten Zugriffen geschützt sein muß.

Der Transaktionsmonitor UTM (SINIX) erfüllt oben genannte Anforderungen und stellt darüber hinaus weitere Funktionen zur Verfügung, so z.B. den Wiederanlauf von kompletten Anwendungen, die Synchronisation von lokalen Datenhaltungssystemen, die Kopplung von Anwendungen im Rechnerverbund, die Verwaltung von Verbindungen und die Nachrichtensteuerung bei lokalen Anwendungen und im Rechnerverbund, das Konfigurieren der Hard- und Softwareumgebung und schließlich die Administration des gesamten Transaktionssystems.

Benutzerschnittstelle

Die UTM-Programmschnittstelle erfüllt die einzige existierende Norm, nämlich die deutsche Norm DIN 66265 KDCS (**K**ompatible **D**aten**c**ommunikations-**S**chnittstelle), ergänzt durch wichtige zusätzliche Aufrufe.

Diese Schnittstellendefinition bestimmt im wesentlichen die Struktur der Teilprogramme, aus denen dann eine Anwendung gebildet wird. Die folgenden Begriffe sind hier von Bedeutung:

Dialogschritt

Ein Dialogschritt beginnt mit einer Dialognachricht und endet mit der Antwort der Anwendung. Eine Transaktion besteht entweder aus einem oder aus mehreren Dialogschritten.

Vorgang

Ein Vorgang ist die Abarbeitung eines in sich geschlossenen Auftrags durch eine Anwendung, bei der ein oder mehrere Teilprogramme durchlaufen werden.

Teilprogramm

Ein Teilprogramm ist ein selbständig übersetzbarer Teil des gesamten Anwendungsprogramms, der normalerweise einen Dialogschritt bearbeitet. Ein Teilprogramm wird durch seinen Transaktionscode (TAC), eine logische Bezeichnung für das Programm, gekennzeichnet.

Transaktion

Das Raster der Sicherungspunkte innerhalb eines Vorgangs bestimmt die UTM-Transaktionen in diesem Vorgang.

Sicherungspunkt

Ein Sicherungspunkt beschreibt einen konsistenten Zustand aller Daten eines Vorgangs und ist zugleich Rücksetzpunkt bei Fehlern und Aufsetzpunkt bei einem Wiederanlauf nach einem Systemausfall. Am Ende eines Vorgangs wird immer ein Sicherungspunkt gesetzt, weitere Sicherungspunkte innerhalb eines Vorgangs sind möglich.

Dialogschritt und Teilprogrammlauf

Aus Sicht der Anwendung beginnt ein Dialogschritt mit dem Empfang einer Nachricht von der Datenstation (Client) und endet mit der Ausgabe der Antwort. Jeder Dialogschritt wird in mindestens einem Teilprogrammlauf bearbeitet. Das heißt, bei Eintreffen einer Nachricht wird ein Teilprogramm gestartet. Sobald das Teilprogramm beendet ist, wird die Antwort an die Station zurückgeschickt oder ein weiteres Teilprogramm aufgerufen.

Transaktion und Vorgang

Eine der wichtigsten Aufgaben von UTM (SINIX) ist es, Benutzerdaten vor Störungen zu schützen. Richtet ein Benutzer einen Auftrag an eine Anwendung, sind bei Update-Vorgängen Modifikationen am zentralen Datenbestand notwendig. Diese Änderungen werden nicht gleichzeitig, sondern zeitlich versetzt durchgeführt. Tritt dabei ein Verbindungsverlust oder eine andere Störung auf, ist ohne TP-Monitor die Konsistenz der Benutzerdaten nicht mehr gewährleistet.

Um dies zu verhindern, werden in UTM (SINIX) logisch zusammengehörige Aufgaben zu Transaktionen zusammengefaßt. Alle innerhalb einer Transaktion stattfindenden Änderungen der Daten werden gemäß der „Alles-oder-Nichts"-Regel entweder vollständig oder gar nicht ausgeführt. Die Transaktion ist in diesem Sinn eine unteilbare Verarbeitungseinheit. Da die Wirkung einer Transaktion auch nach einer abnormalen Beendigung erhalten bleiben muß, wird sie bei Transaktionsende gesichert. Das Transaktionsende heißt daher auch Sicherungspunkt. Die Aktionen einer nicht erfolgreich abgeschlossenen Transaktion werden, einschließlich der DB-Transaktionen, durch UTM (SINIX) zurückgesetzt. Insgesamt ist eine Transaktion durch folgende Eigenschaften gekennzeichnet, die unter dem Akronym ACID (**A**tomicity, **C**onsistency, **I**solation, **D**uration) zusammengefaßt werden:

Unteilbarkeit (atomicity)

Die von einer Transaktion betroffenen Änderungen der Anwenderdaten werden entweder vollständig oder gar nicht ausgeführt.

Konsistenz (consistency)

Die Anwenderdaten sind jederzeit in einem konsistenten Zustand.

Isolierung (isolation)

Die Änderung von Daten, die in einer Transaktion durchgeführt wird, wird erst bei Transaktionsende für andere Benutzer sichtbar.

Dauerhaftigkeit (duration)

Ist eine Transaktion abgeschlossen, so ist ihre Wirkung dauerhaft.

Zurücksetzen einer Transaktion und Koordination mit Datenbank-Transaktionen

Es gibt verschiedene Gründe, warum eine Transaktion zurückgesetzt werden muß, z.B. durch Anforderungen aus einem Teilprogramm, wegen vorzeitiger Beendigung der Anwendung, nach Verbindungsverlust, als Folge von Programmfehlern oder aus Systemgründen. Wird eine Transaktion zurückgesetzt, werden Speicherbereiche im Verarbeitungsrechner auf den Stand zurückgesetzt, den sie zu Beginn der Transaktion hatten. Globale Speicherbereiche werden auf den Stand gebracht, den sie unmittelbar vor dem ersten Zugriff in der Transaktion hatten.

Die Daten eines mit UTM(SINIX) gekoppelten Datenbanksystems zählen ebenfalls zu den Anwenderdaten und unterliegen deshalb dem Transaktionsprinzip. Um die Transaktion als unteilbare Einheit zu erhalten, ist eine Koordination zwischen dem Transaktionsmonitor und dem Datenbanksystem notwendig. Soll eine Datenbank-Transaktion durch einen Aufruf im Teilprogramm beendet werden, wird die Beendigung im Datenbanksystem verzögert, bis auch die UTM-Transaktion beendet wird. Auch im Fall des Wiederanlaufs einer Anwendung findet eine Koordination mit dem Datenbanksystem statt.

Wiederanlauf eines Vorgangs und einer Anwendung

Wird ein aus mehreren Transaktionen bestehender Dialog unterbrochen, kann dieser Vorgang fortgesetzt werden. Befindet sich der Vorgang zum Zeitpunkt der Unterbrechung nicht auf einem Sicherungspunkt, sondern innerhalb einer Transaktion, wird die Transaktion zurückgesetzt und der Vorgang am zuletzt erreichten Sicherungspunkt fortgesetzt. Wurde eine Anwendung abnormal beendet, führt UTM (SINIX) automatisch den Wiederanlauf einer Anwendung durch. Die Daten der Anwendung werden auf den zuletzt erreichten konsistenten Zustand gesetzt.

Anwendungen

In einem Rechner können unter der Kontrolle von UTM (SINIX) mehrere voneinander unabhängige Anwendungen existieren. Jede dieser Anwendungen wird gesondert generiert und administriert. Der Betrieb kann somit unterschiedliche Aufgaben, wie z.B. Lagerhaltung, Personalwesen, usw. in vollständig getrennten Anwendungen realisieren. Jeder Benutzer meldet sich bei der gewünschten Anwendung an und arbeitet ab diesem Zeitpunkt ausschließlich mit dieser Anwendung. Dabei ist es für die Anwendung ohne Bedeutung, wieviele Benutzer gleichzeitig mit

dieser Anwendung arbeiten. Datenstationen und Anwendungen im gleichen oder einem anderen Rechner können logische Verbindungen aufbauen und Aufträge absenden. Eine Anwendung kann auch von sich aus Verbindungen zu anderen Kommunikationspartnern im Netz einrichten und Nachrichten verschicken.

Für jede Datenstation, die mit der UTM-Anwendung arbeitet, existiert ein eigener Dialogprozeß, der **D**ialog**t**erminal**p**rozeß (DTP) genannt wird. Er wird entweder durch expliziten Aufruf über die Shell oder automatisch nach erfolgreicher Anmeldung des Datenstationsbenutzers an das SINIX-System gestartet. Für Ausgaben an Drucker werden spezielle Druckprozesse verwendet, die für jeden angeschlossenen Drucker eingerichtet werden.

Die Netzanbindung der Anwendung erfolgt über getrennte Netzprozesse, die sowohl den Verbindungsauf- bzw. -abbau, als auch den Datenaustausch zwischen den beteiligten Partnern abwickeln. Pro Verbindung wird ein eigener Prozeß verwendet, der bis zum Abbau dieser Verbindung existiert.

Eine UTM-Anwendung wird durch Generierung mit Hilfe eines Dienstprogramms erstellt. Hierbei wird auch die von einer Anwendung benötigte Konfiguration festgelegt. Sie enthält u.a. folgende Informationen:

- Transaktionscodes und deren Eigenschaften,
- Anwendungsteilprogramme und deren Eigenschaften,
- Datenendgeräte (Drucker und Terminals) und deren Verwendungsart,
- Benutzernamen und deren Berechtigung,
- verteilte Anwendungen und deren Eigenschaften.

Gesteuert über weitere Parameter wird ein ablauffähiges Anwendungsprogramm erzeugt, das neben den Anschlußroutinen an UTM (SINIX) die Anwendungsteilprogramme enthält. Dieses Gesamtprogramm wird mittels einer SINIX-Startprozedur als Hintergrundprozeß (der sogenannte Mainprozeß) gestartet. Danach werden von diesem Mainprozeß so viele Workprozesse aktiviert, wie in den Startparametern festgelegt wurde.

Verteilte Transaktionsverarbeitung mit UTM

Die verteilte Transaktionsverarbeitung ist bereits seit der ersten Freigabe im Jahr 1988 integraler Bestandteil von UTM (SINIX). Mit dieser Funktionskomponente wird die rechnerübergreifende Transaktionsverarbeitung ermöglicht. Neben Datenstationen werden dafür auch ferne UTM-Anwendungen als Dialogpartner zugelassen, um mit Hilfe von Dialognachrichten Unteraufträge starten und Ergebnisse empfangen zu können.

Da Unteraufträge durch entfernte Anwendungen in Form von Transaktionen abgewickelt werden, liegen bei der verteilten Transaktionsverarbeitung ebenso wie bei einer DB/DC-Transaktion geschachtelte Transaktionen mit der Notwendigkeit der Synchronisierung vor. Der Unterschied besteht nur darin, daß Auftrag und Ergebnis als Dialognachrichten übermittelt werden und zur Synchronisation geeignete Kommunikationsprotokolle (das sogenannte Two-Phase-Commit-Protokoll) verwendet werden.

Die verteilte Transaktionsverarbeitung bietet auch die Möglichkeit, zwischen zwei UTM-Anwendungen parallele Verbindungen aufzubauen und mehrere Aufträge (auch Dialogaufträge) gleichzeitig abwickeln zu können.

Eine UTM-Anwendung kann über verteilte Transaktionsverarbeitung mit einer Partneranwendung entweder synchron kommunizieren, d.h. eine Anwendung sendet einen Dialogauftrag (Nachricht) an den Partner und wartet auf die Antwort, oder aber die Kommunikation erfolgt asynchron, also ohne direkte Antwort.

Die verteilte Transaktionsverarbeitung in UTM (SINIX) ermöglicht die Transaktionsverarbeitung von UTM-Anwendungen auch im Netzverbund von SINIX- und BS2000-Systemen. Wegen der Verwendung von Protokollen gemäß LU6.1 (IBM SNA) zur Kommunikationssteuerung ist eine gesicherte Verarbeitung zwischen den Transaktionssystemen UTM (SINIX) und CICS bzw. IMS von IBM möglich. Derzeit in Entwicklung befindet sich die Implementierung von OSI-TP-Protokollen. Mit Hilfe dieser inzwischen unter ISO 10026 weltweit gültigen Norm wird eine Transaktionsverarbeitung mit Systemen aller Partner möglich, die diesen Standard ebenfalls realisiert haben.

Die derzeit verfügbaren bzw. in Vorbereitung befindlichen Kopplungsmöglichkeiten über UTM (SINIX) sind in Abbildung 8-2 dargestellt.

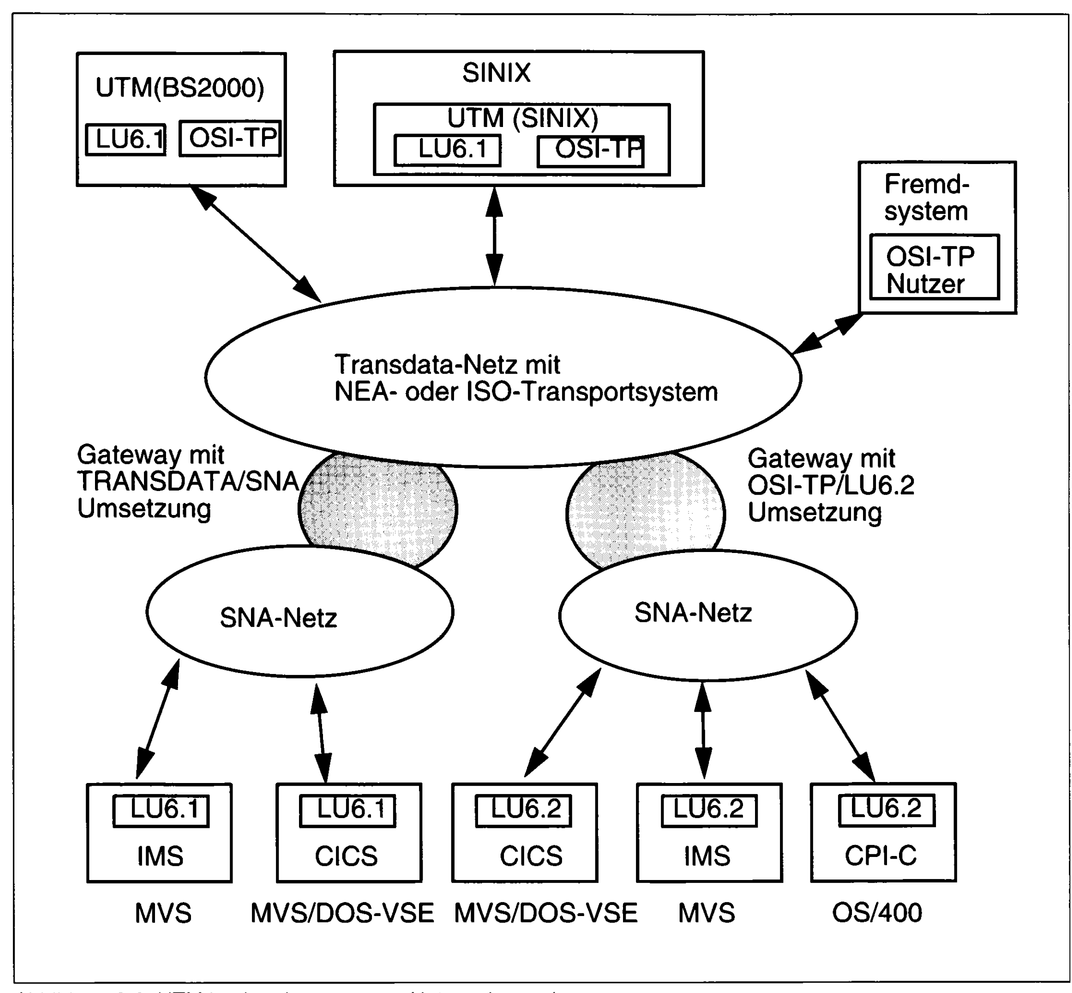


Abbildung 8-2: UTM in einer heterogenen Netzwerkumgebung.

Die Client-Server-Architektur mit UPIC und UTM (SINIX)

Ein wichtiges Kriterium für die erfolgreiche Handhabung der Schnittstelle zwischen Mensch und Maschine ist die anwendungsfreundliche Strukturierung der Bedienoberfläche (Präsentationsschicht). Hierfür werden immer mehr die Möglichkeiten von Produkten wie OSF/Motif und MS-Windows aus der SINIX- und der MS-DOS-Welt eingesetzt. Da die Bildschirmgestaltung eigentlich nicht Aufgabe eines Transaktionsmonitors wie UTM (SINIX) ist, wird die Bildschirmsteuerung zunehmend aus den bisherigen UTM-Anwendungsprogrammen in sogenannte Clients ausgelagert. Die eigentliche Verarbeitungsleistung wird als Server in den jeweiligen UTM-Anwendungen erbracht.

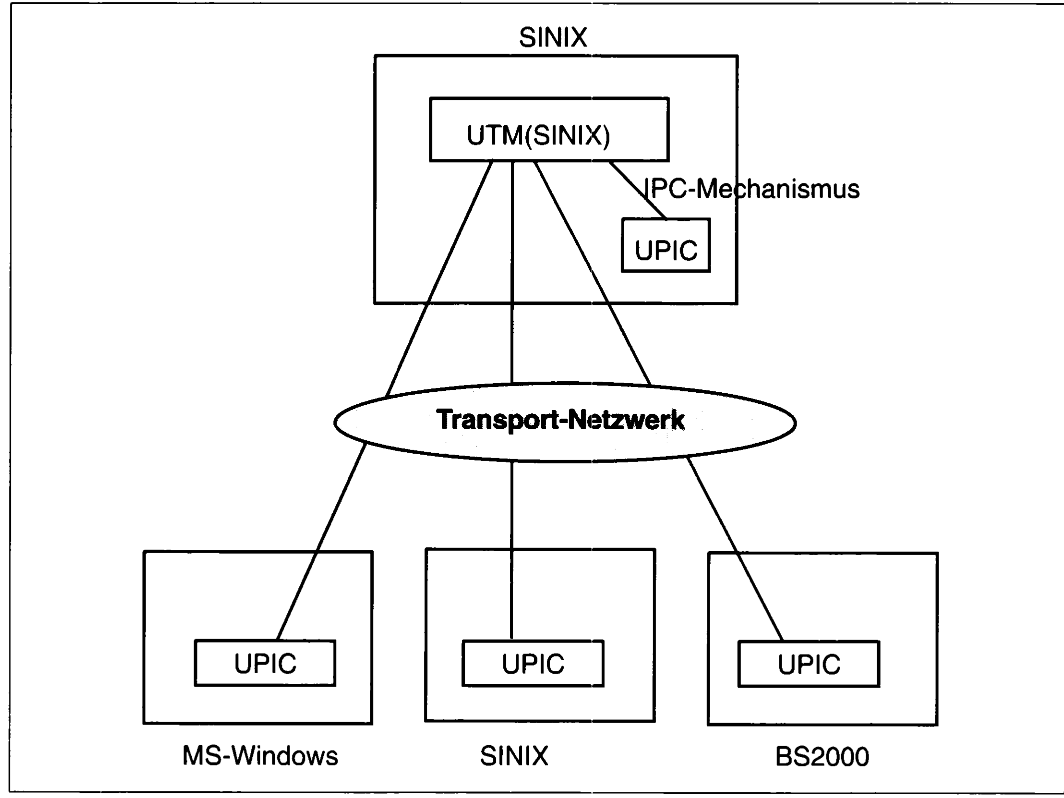


Abbildung 8-3: Client-Server-Kommunikation mit UPIC und UTM (SINIX).

Das Client-Server-Konzept hat zum Ziel, den Nutzern eines Rechennetzes die Ressourcen des gesamten Verbundes (also Daten, Programme und Geräte) verfügbar zu machen. Ein Client-Server-Konzept ist immer dann sinnvoll einzusetzen, wenn viele Anforderer (Clients) vorhanden sind, die dieselben Dienstleistungen (Server) benötigen. Hierzu wurde das Produkt UPIC entwickelt, das eine Programmschnittstelle gemäß der X/Open-Spezifikation CPI-C (**C**ommon **P**rogramming **I**nterface for **C**ommunication) darstellt. Es ermöglicht die Kopplung von UPIC-Anwendungen mit UTM-Anwendungen, wie z.B. den Anschluß von grafischen Oberflächensystemen. UPIC wurde auf alle in Betracht kommenden Systemlinien portiert und ist auch für MS-Windows und BS2000 verfügbar. Das prinzipielle Zusammenwirken von UPIC-Clients mit UTM-Servern in SINIX und BS2000 ist in Abbildung 8-3 dargestellt.

Ausblick

Im Vordergrund der Entwicklungen zur Ergänzung des Funktionsumfangs von UTM (SINIX) steht die Unterstützung neuer Standards in Bezug auf Kommunikationsprotokolle und Benutzerschnittstellen. Die Implementierung des OSI-TP-Standards ist gegenwärtig in Arbeit und wird mit der nächsten Version von UTM (SINIX) freigegeben. Über einen derzeit in Entwicklung befindlichen Protokollumsetzer OSI-TP/LU6.2 kann dann auch mit Anwendungen kommuniziert werden, die das LU6.2-Protokoll von IBM anwenden.

Um die weitere Verbreitung von UTM (SINIX) in der Welt der offenen Systeme zu fördern, ist es notwendig, die Benutzerschnittstelle KDCS (deutscher Standard) durch international festgelegte Schnittstellen zu erweitern. Dazu werden in erster Linie die Festlegungen von X/Open Verwendung finden. Als erster Schritt wurde bereits die gesicherte Kommunikation mit Datenbanksystemen (Ressourcenmanager) über die Systemschnittstelle XA realisiert, und mit INFORMIX und einer Pilotversion von ORACLE V7 erfolgreich getestet. Nach Verabschiedung weiterer Standards gemäß des X/Open-Referenzmodells für Transaktionsverarbeitung sollen diese ebenfalls von UTM (SINIX) unterstützt werden.

Datenbanksysteme

INFORMIX

INFORMIX steht für eine weitverzweigte Familie von Datenbankprodukten, die für eine breite Palette von Rechnerarchitekturen, einschließlich der Rechner von Siemens Nixdorf, verfügbar ist. Neben dem eigentlichen DBMS (**Data**base **M**anagement **S**ystem) umfaßt die INFORMIX-Produktfamilie auch das ganze Spektrum moderner Datenbankwerkzeuge und -schnittstellen sowie Komponenten zur verteilten Datenbankverarbeitung nach dem neusten Stand der Entwicklung.

INFORMIX ist das im UNIX-Markt am weitesten verbreitete Datenbankprodukt. Mit einer Durchdringung des SINIX-Maschinenparks von etwa 50% beweist das Produkt seine hervorragende Eignung und Einsatzfähigkeit. Nach dem Erwerb der Sourcelizenz für INFORMIX-Produkte hat Siemens Nixdorf ein Team von Experten aufgebaut, das die Portierung auf Siemens-Nixdorf-Systeme durchführt und Kunden bei Problemfällen sachkundig unterstützt.

Aufgrund dieser strategischen Partnerschaft zwischen Siemens Nixdorf und INFORMIX Software, Inc. konnten einige gemeinsame Entwicklungsvorhaben erfolgreich abgeschlossen werden. Das hat zu einer stetigen Kompetenzsteigerung bei Siemens Nixdorf auf diesem Gebiet geführt. Dies wiederum stellt eine gute Grundlage für die Abstimmung und Realisierung weiterer gemeinsamer Entwicklungsziele dar.

Im folgenden werden einige Produkte aus der INFORMIX-Familie kurz vorgestellt.

INFORMIX-OnLine

INFORMIX-OnLine ist ein leistungsstarker, SQL-basierter relationaler Datenbankserver für OLTP, mit hoher Verfügbarkeit, hoher Verarbeitungsleistung unter Gewährleistung der Datenintegrität sowie Multi-Media-Eigenschaften unter Einhaltung des SQL-Standards (ANSI Level 2, 1989).

INFORMIX-SE

INFORMIX-SE ist ein einfach zu nutzendes SQL-basiertes relationales Datenbankmanagementsystem, das für den Einsatz in kleinen und mittleren Anwendungen gedacht ist und ohne Datenbankadministrator betrieben werden kann.

INFORMIX-TP/XA

INFORMIX-TP/XA verbindet INFORMIX-Online mit Transaktionsmanagern auf Basis der von X/Open standardisierten XA-Schnittstelle. Hiermit können globale Transaktionen auch über mehrere heterogene Computer- und Datenbanksysteme hinweg für eine Vielzahl von parallelen Nutzern koordiniert werden.

C-ISAM

C-ISAM ist sozusagen das klassische ISAM-Produkt in der UNIX-Welt und bietet dem C-Programmierer Werkzeuge zur Speicherung von Daten nach dem XPG4-Standard von X/Open.

INFORMIX-OnLine/FMWORM

Mit INFORMIX-OnLine/FMWORM wurde das konventionelle Konzept eines Datenbankmanagementsystems um die Fähigkeit erweitert, Bilddateien oder große Mengen an Binärdaten (BLOBs) als Teil der relationalen Datenbank abzuspeichern. Mit INFORMIX-OnLine/FMWORM wurde die Möglichkeit geschaffen, die Speicherung dieser Daten auf dem optischen Subsystem FMWORM, einer Entwicklung von Siemens Nixdorf, zu integrieren.

ESQL-C, ESQL-COBOL

Datenbankzugriffe aus herkömmlichen Programmiersprachen der dritten Generation sind über Precompiler-Produkte möglich, die sowohl eine statische als auch eine dynamische Einbindung der SQL-Anweisungen in normale C- bzw. COBOL-Programme erlauben.

INFORMIX-SQL

INFORMIX-SQL bietet dem Anwender den interaktiven Zugriff unter einer menügesteuerten Bedienoberfläche auf der Basis von SQL-Eingaben. Bestandteil des Produkts sind ein Listengenerator und ein Maskengenerator. Beide unterstützen die interaktive und die programmgesteuerte Nutzung. Mit Hilfe des Maskengenerators kann man sehr schnell einfache menügesteuerte Anwendungen entwickeln. Der Listengenerator bietet bei der Aufbereitung von Ergebnissen in Listenform höchste Funktionalität.

INFORMIX-4GL

INFORMIX-4GL/RDS/ID

Die 4GL-Produkte von INFORMIX bieten eine komplette Entwicklungsumgebung (Compiler, Interpreter, Debugger) für die Entwicklung von Anwendungen auf Basis leistungsstarker Programmiersprachen der vierten Generation. Zusammen mit prozeduralen Sprachelementen sind Datenbankzugriff, Maskenerzeugung und Listengenerierung in einer homogenen Programmiersprache integriert, was zu einer erheblichen Produktivitätssteigerung bei der Anwendungsentwicklung beiträgt.

INFORMIX-4GL++

Die neuen Produkte der 4GL-Familie beinhalten insbesondere Erweiterungen zur Bedienung grafikfähiger Workstations und funktionale Erweiterungen für die objektorientierte Anwendungsentwicklung.

INFORMIX-4GL for ToolBus

INFORMIX-4GL for ToolBus bietet eine leistungsfähige und einfach zu bedienende grafische Entwicklungsumgebung für die Entwicklung von Anwendungen mit den INFORMIX-4GL-Produkten. Das Produkt ist in INFORMIX-OpenCase/ToolBus integriert.

INFORMIX-Connectivity-Lösungen

Netzwerke spielen heute eine wichtige Rolle bei der Einbindung der verschiedenen Informationsbasen innerhalb eines Unternehmens auf PC-, Abteilungs- oder Unternehmensebene. Deshalb sind Produktlösungen gefragt, die den lokalen, aber auch den globalen Zugriff auf alle Informationen direkt vom Arbeitsplatz aus ermöglichen.

Mit dem Produkt INFORMIX-NET wird der entfernte Zugriff über heterogene Rechner auf Basis aller wesentlichen standardmäßigen Netzwerkprotokolle unterstützt. Durch die Transparenz der Datenbanklokalisation eignet sich INFORMIX-NET bestens für den Einsatz von Client-Server-Lösungen in Netzwerken.

INFORMIX-STAR ist das Produkt von INFORMIX für den Datenbankzugriff in verteilter Verarbeitung. Es erlaubt dem Anwender Schreib- und Lesezugriffe auf mehrere Datenbanken auf verschiedenen Rechnern und garantiert gleichzeitig die Datenkonsistenz. Für das Anwendungsprogramm erscheint dabei der ferne Zugriff mit dem lokalen Zugriff identisch. INFORMIX-STAR regelt den Datenverkehr sowohl über LANs als auch über WANs.

INFORMIX-Net-PC ermöglicht den Zugriff von DOS-Anwendungen auf die INFORMIX-Datenbanken im Netz. Die bereitgestellten Entwicklungswerkzeuge 4GL, ESQL und SQL unterstützen den Anwender auf vielfältige Art und Weise bei der Programmentwicklung.

Darüberhinaus wird an Lösungen zur Unterstützung von heterogenen verteilten Anwendungen gearbeitet. INFORMIX ist ein führendes Mitglied der SQL-Access-Group und entwickelt gegenwärtig ein Produkt zur Anbindung von INFORMIX an die IBM-Welt auf Basis von DRDA (**D**istributed **R**elational **D**atabase **A**rchitecture).

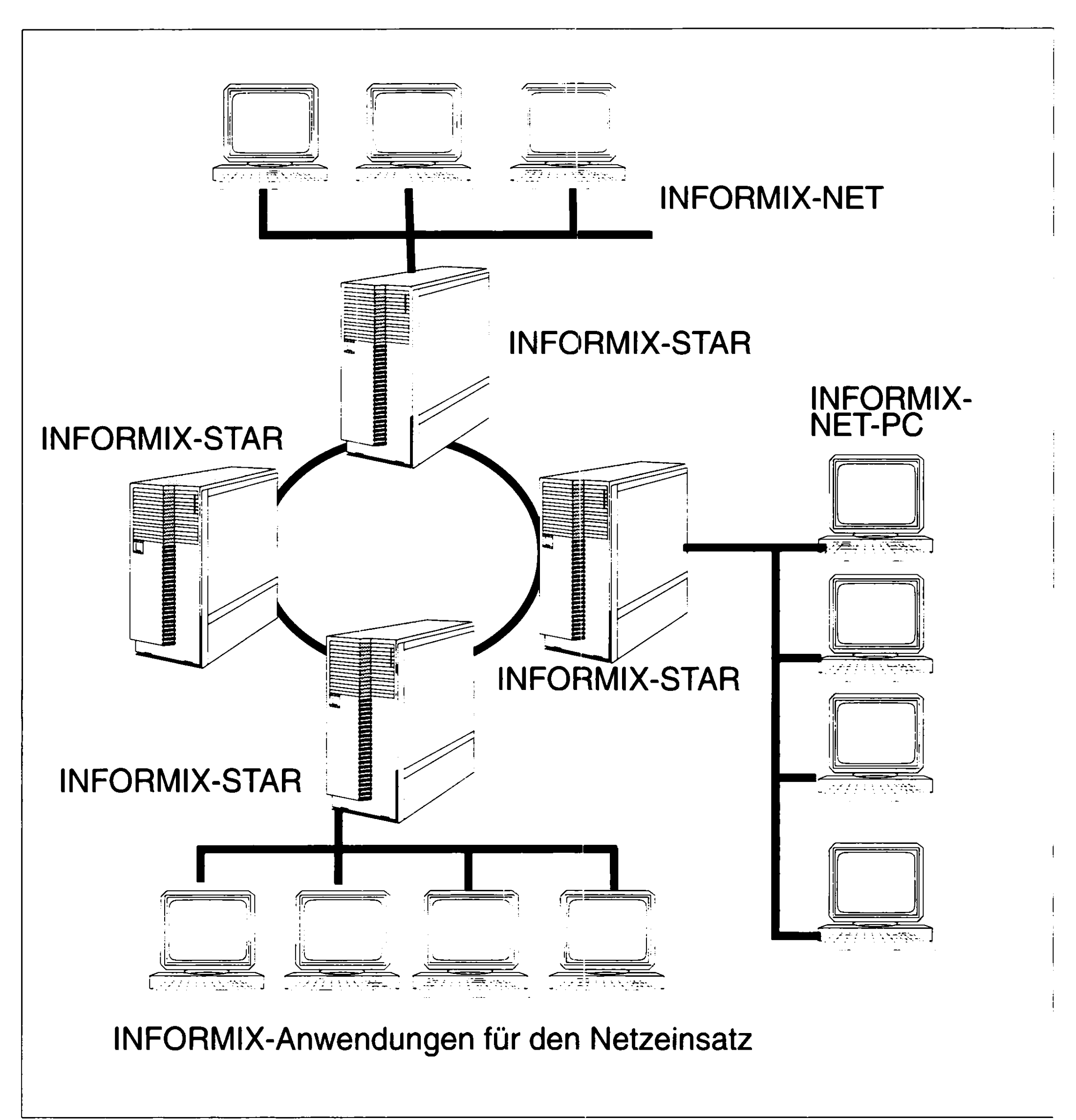


Abbildung 8-4: INFORMIX-NET und INFORMIX-STAR erlauben den fernen und verteilten Zugriff auf INFORMIX-Datenbanken.

Oracle

Oracle ist mehr als nur ein relationales Datenbanksystem, es ist vielmehr eine abgerundete Produktfamilie mit leistungsfähigen Werkzeugen zur Erstellung von Datenbanken und Anwendungen, zum komfortablen interaktiven Zugriff auf Daten und mit Gateways zur Anbindung an Fremdsysteme und anderes mehr. Die herausragenden Leistungsmerkmale der Oracle-Produktfamilie sind:

- eine allen Standards konforme SQL-Schnittstelle,
- konsequente Client-Server-Architektur,
- die Ablauffähigkeit auf allen relevanten Systemplattformen und dadurch nahezu unbeschränkte Portabilität und Konnektivität.

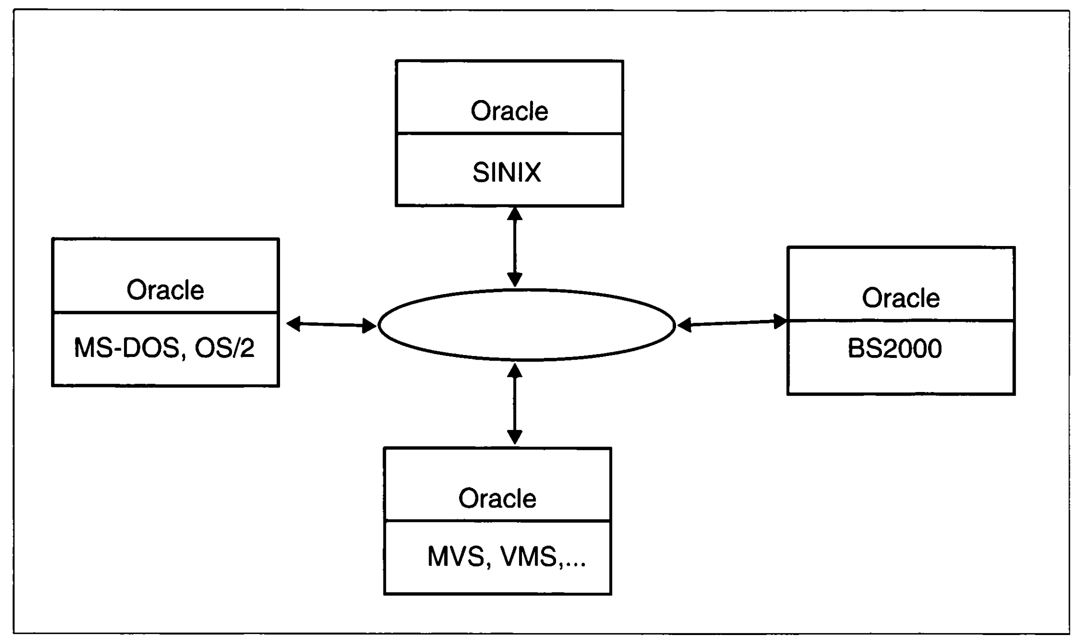


Abbildung 8-5: Oracle ist auf unterschiedlichen Systemen ablauffähig, die über LAN oder WAN miteinander verbunden sind.

Die wichtigsten Oracle-Produkte werden im folgenden kurz beschrieben.

Oracle DBMS

Oracle DBMS ist ein leistungsfähiges, relationales Datenbanksystem mit breit angelegtem Funktionsspektrum; es eignet sich für alle Anwendungstypen von der komplexen Entscheidungsunterstützung bis hin zum klassischen OLTP-Betrieb.

SQL*Net

SQL *Net ist ein Baustein für ortstransparente Zugriffe auf verteilte Datenbanken, wobei alle gängigen Netzwerkprotokolle unterstützt werden.

SQL*Connect

SQL *Connect bildet den Gateway zu anderen Datenbanksystemen.

Precompiler

Hier handelt es sich um einen Precompiler für Embedded SQL in C, COBOL, FORTRAN und PL/1.

PL/SQL

Dies ist eine Sammlung Datenbankprozeduren mit Kontrollstrukturen und anderen Leistungsmerkmalen in einer höheren Programmiersprache.

XA-Bibliothek

Die XA-Bibliothek bietet einen Transaktionsmanager, basierend auf der von X/Open festgelegten XA-Systemschnittstelle. Unter Verwendung der XA-Schnittstelle können globale Transaktionen innerhalb mehrerer heterogener Computersysteme und Datenbanken für eine große Zahl von parallel arbeitenden Anwendern koordiniert werden.

SQL*Forms und SQL*Menu

Dies sind Anwendungsgeneratoren für anspruchsvolle Bildschirmanwendungen.

SQL*Plus und SQL*ReportWriter

Dies ist eine Interaktive SQL-Schnittstelle bzw. ein Werkzeug zur Listenerstellung.

CASE*Dictionary, CASE*Designer und CASE*Generator

Werkzeuge für Entwurf und Erstellung von Oracle-Datenbanken und -Anwendungen sowie zur Strukturierung von Abläufen.

DRIVE - „Fourth Generation Language" von Siemens Nixdorf

Das Softwareprodukt DRIVE von Siemens Nixdorf umfaßt die Fourth **G**eneration **L**anguage (4GL) DRIVE und das zugehörige Entwicklungssystem. Mit DRIVE werden OLTP-Anwendungen entwickelt, die SESAM-, UDS-, INFORMIX sowie zukünftig auch ORACLE- und DB2-Datenbanken verwenden.

DRIVE bietet einen hohen Sprachkomfort. Dazu gehören

- in die Sprache integrierte SQL-Anweisungen für die Definition von Datenbanken (DDL), den Datenbankzugriff sowie für die Definition von temporären Sichten (Views).
- leistungsfähige Anweisungen für Bildschirmformate auf Alpha- und Grafikterminals.
- umfassende Anweisungen für die Reporterstellung.
- Sprachmittel und Automatismen zur Transaktionssteuerung und Client-Server-Konfiguration.
- Funktionen, die es ermöglichen, internationalisierbare Anwendungen auf der Grundlage des NLS (**N**ative **L**anguage **S**ystem) zu erstellen.

Das DRIVE-System besitzt eine einheitliche Schnittstelle zu den von ihm unterstützten Datenbanksystemen auf der Grundlage des ISO-SQL-Standards. SQL als Abfragesprache ist voll in die DRIVE-Sprache integriert. DRIVE-Anwendungen sind sowohl im Einplatzbetrieb als auch unter Steuerung eines Transaktionsmonitors (UTM) ablauffähig, wobei für einen Wechsel zwischen den beiden Betriebsarten keinerlei Änderungen am DRIVE-Programm nötig sind. Eine Anwendung kann auch ohne Änderung am DRIVE-Programm auf einem BS2000- oder einem SINIX-

System eingesetzt werden. So können DRIVE-Programme, die auf einem BS2000-System ablaufen sollen, auch mit dem grafischen Entwicklungssystem in SINIX erstellt und getestet werden.

Damit sind DRIVE-Programme weitgehend unabhängig von dem zugrundeliegenden Datenbank- und Betriebssystem. Sie werden sich daher ihre Einsatzfähigkeit und ihren Wert auf lange Sicht hin erhalten.

Das Repository-System ERMS

Das Repository-System ERMS (**E**nterprise **R**epository **M**anagement **S**ystem) ist ein spezialisiertes Datenbanksystem für die Verwaltung von Metadaten (beschreibende Daten über Daten). Die informationstechnischen Beschreibungen eines Unternehmens und seiner Datenverarbeitungssysteme ist üblicherweise in vielfältigen Ablagen, vom Aktenordner bis zur Datenbank, verstreut. In einem Unternehmen muß dann versucht werden, die Qualität der Daten wie z.B. Aktualität, Konsistenz und Redundanzfreiheit durch organisatorische Vorschriften und Kontrollen zu erreichen. In einem Repository werden dagegen alle diese Informationen konsistent und einheitlich verwaltet: Werkzeuge für Unternehmensmodellierung und Anwendungsentwicklung (z.B. Editoren, Generatoren), für den Systembetrieb (z.B. System- und Netzmanagement) sowie für Konfigurationsverwaltung, Projektmanagement, Installation und Wartung können ihre beschreibenden Daten über die Schnittstelle IDDS im Repository-Management-System ERMS ablegen, modifizieren, lesen und löschen. Soweit die Informationen von übergreifender Bedeutung sind, muß die Struktur der Daten, also das Informationsmodell, zwischen den Repository-Anwendungen abgestimmt werden.

Ein einheitliches Datenverwaltungssystem sowie ein redundanz- und widerspruchsfreies Informationsmodell gewährleisten die Qualität der Daten. Gleiche Sachverhalte können nicht mehrfach oder widersprüchlich eingegeben werden. Entwicklungswerkzeuge können direkt auf die Ergebnisse der vorangegangenen Entwicklungsstufe zugreifen. Die einheitliche Modellierung nach dem Entity-Relationship-Modell gestattet globale Recherchen und das Erkennen von Abhängigkeiten. ERMS wird so zur Drehscheibe für sämtliche beschreibende Unternehmensinformationen, für Softwareentwicklung, -betrieb und -pflege.

Die Informationsmodelle, die dem Repository zugrunde liegen, können vom Anwender im Rahmen der Entity-Relationship-Modellierung frei gestaltet werden. Dadurch kann ERMS in eine spezifische Anwenderumgebung eingebunden werden. Durch Export/Import-Funktionen (auch für Schemata möglich) läßt sich ERMS mit anderen Data-Dictionary/Repository-Systemen zusammenschließen.

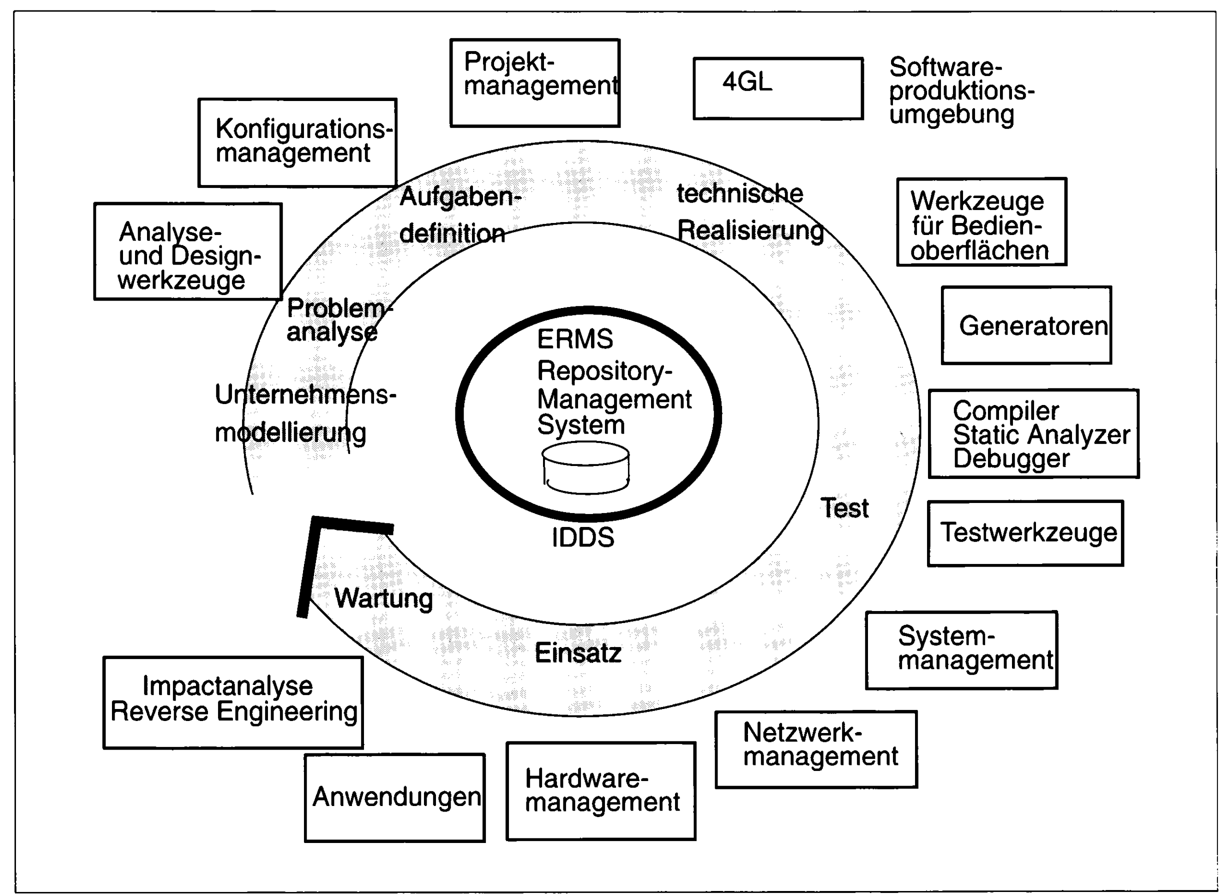


Abbildung 8-6: Das Repository ERMS enthält Informationen über den gesamten Lebenszyklus von Softwareprodukten.

Über die Schnittstelle IDDS kann der Anwender nicht nur seine eigenen Programme auf ERMS aufsetzen. Siemens Nixdorf setzt selbst Produkte darauf auf, wie z.B. das **O**pen **S**ystem **C**ASE **E**nvironment (OSCE) oder das TRANSDATA-Communication-Network-Management.

Bei den vielfältigen parallelen Zugriffen auf das Repository werden Datensicherheit und Konsistenz dadurch gewährleistet, daß ERMS seinerseits auf den bewährten Funktionen des relationalen Datenbanksystems INFORMIX aufsetzt. Unstrukturierte, umfangreiche Textinformationen (z.B. Sourcecode, Dokumentation) werden von ERMS im SINIX-Dateisystem abgelegt. Sie sind dabei voll in das Sicherheits- und Transaktionskonzept einbezogen.

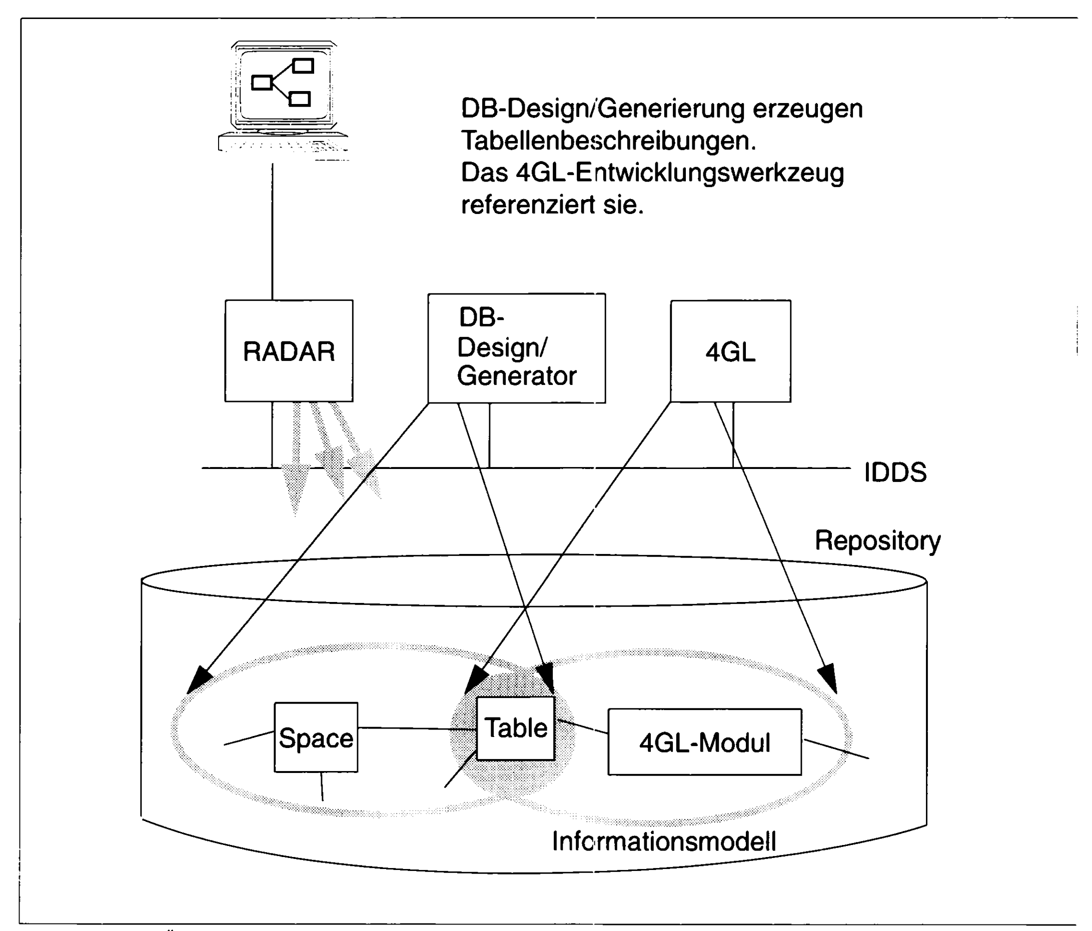


Abbildung 8-7: Überlappende Informationsmodelle im ERMS-Repository und Auswertung über RADAR.

Das Entity-Relationship-Modell ermöglicht eine einfache und intuitive Modellierung der Wirklichkeit. Der Anwender muß sie nicht in relationale Strukturen übersetzen, denn das wird vom System selbst erledigt. Über die grundlegenden Modellelemente Entity-Typ, Relationship-Typ und Attribute hinaus können vielfältige

Konsistenzbedingungen für Beziehungen und Attribute definiert werden, die vom System auch überwacht werden. Die Einhaltung liegt dabei also nicht in der Verantwortung der einzelnen Anwendungen.

Da die Informationselemente, z.B. bei der Softwareentwicklung oder beim Konfigurationsmanagement, im Lauf der Zeit Veränderungen unterworfen sind, unterstützt ERMS ein Versionskonzept für Entities. Deren Lebenszyklus, z.B. in Bezug auf Qualität und/oder Phase im Entwicklungsprozeß, kann durch Partitionen im Repository modelliert werden, welche die Entities dann durchwandern.

Neben den Funktionen zur Eingabe und Manipulation bietet ERMS leistungsfähige Funktionen für die Wiedergewinnung von Informationen, die sich am hohen technischen Stand der Datenbanktechnik orientieren. Die Möglichkeit, Suchabfragen mit vielfältigen Bedingungen zu koppeln oder sie auf ganze Beziehungspfade auszudehnen sowie mengenorientierte Operationen machen den Zugriff besonders effektiv. Für den interaktiven Zugriff steht neben der Kommandoschnittstelle die formatorientierte Ein-/Ausgabe ERMS-easy-access sowie der grafische Browser RADAR zur Verfügung. Mit RADAR können Repository-Schema und -Inhalt als Bäume, Netze, Tabellen oder Diagramme dargestellt werden.

Die Programmschnittstelle IDDS ist als universelle, systemunabhängige Schnittstelle konzipiert. Ein verteiltes ERMS mit Client-Server-Architektur bietet über die identische IDDS-Schnittstelle Zugriff von SINIX-, BS2000- und PC-Systemen auf einen SINIX-Repository-Server.

Wegen der zentralen Bedeutung der Repository-Schnittstelle ist hier eine Standardisierung wünschenswert. ERMS entspricht heute weitgehend dem IRDS-Standard des American National Standards Institute (ANSI). Zukünftig wird ERMS auch eine Schnittstelle zur Objektverwaltung gemäß ECMA-PCTE anbieten, sobald diese sich als De-facto- oder internationaler Standard durchsetzt.

INFORMIX hat von Siemens Nixdorf eine Lizenz für das ERMS-Repository-System erworben und vermarktet ERMS nun erfolgreich für große OLTP-Systeme, als CASE-Repository und als Technologie, die wiederum von Drittfirmen zur Entwicklung zusätzlicher Anwendungen in Lizenz erworben werden kann.

PC-Integration

SQL-Gateways

Siemens Nixdorf bietet auf PC-Ebene innerhalb des OCIS-Bürosystems die ComfoWare-Produktfamilie als geeignete Bürolösung an.

Das Produkt ComfoBase ist das Datenbanksystem innerhalb von ComfoWare und leitet sich von den SQL-Produkten der Gupta Technologies, Inc. ab. Siemens Nixdorf und Gupta Technologies, Inc. sind OEM-Partner. Neben der für ein PC-Produkt äußerst leistungsfähigen, relationalen Funktionaliät ist die Möglichkeit der Verbindung zu anderen Datenbanksystemen auf der Grundlage von Gatewayprodukten eine wichtige Eigenschaft der Produkte von Gupta. Dieses Konzept wurde durch den konsequenten Einsatz der Client-Server-Architektur verwirklicht, bei dem ein SINIX-Rechner die Rolle des zentralen Serversystems für die DOS-Anwendungen übernimmt, wodurch der Zugriff auf INFORMIX- (Serverrechner) bzw. SESAM- oder UDS-Datenbanken (Gatewayrechner) ermöglicht wird.

Die Software-Produkte SQL Gateway/INFORMIX und SQL-Gateway/Mainframe (SESAM and UDS) werden derzeit auf Grundlage der Gupta-Gateway-Produkte portiert bzw. entwickelt.

Die SQL-Gateway-Produkte werden auf einem SINIX-System installiert, das als Server-Rechner fungiert. Die Gateways haben die folgenden wesentlichen Aufgaben:

- Konzentratorfunktionen
- Protokollkonverter (SQL, Call-Level, Transport)
- Datentypanpassung
- Fehlerbehandlung

Die ersten Versionen der beiden Produkte dieser Familie wurden 1992 freigegeben. Das Ziel weiterer Entwicklungen wird neben der Portierung der Produkte auf andere Plattformen auch eine Funktionserweiterung und Anpassung an neue DBMS-Versionen sein.

MMC

Das Software-Produkt MMC (**M**icro-**M**ainframe-**C**onnection) wurde für SINIX in der Version 3.1 und für DOS in der Version 3.0 freigegeben. Das wichtigste Ziel von MMC ist es, den Zugriff auf BS2000-Datenbankmanagementsysteme (SESAM, UDS, GOLEM, INFPLAN, ADILOS, EASY, DVS) durch eine bequeme, automatisierte Verbindung zu den verschiedensten Anwendungen in SINIX- und DOS-Systemen zu erweitern. MMC Version 3.1 für SINIX kann darüber hinaus als Werkzeug für die Unterstützung von interaktiven PC-Host-Verbindungen genutzt werden, in denen PCs als intelligente Arbeitsplätze eingesetzt werden. MMC ermöglicht SINIX-Systemen auch den Zugang zur IBM-Welt und damit zu den Datenbanken unter MVS.

UPIC für MS-Windows

UPIC ermöglicht die Kommunikation zwischen MS-Windows und UTM-Anwendungen auf SINIX- oder BS2000-Systemen. Mit diesem Produkt können auch PCs Anwendungen im Rahmen der Transaktionsverarbeitung benutzen.

9 Verfügbarkeit, Sicherheit und Qualität in SINIX

Hohe Systemverfügbarkeit

Verfügbarkeit ist eine Systemeigenschaft, die in den letzten Jahren vor allem im Bereich der offenen Systeme an Bedeutung gewonnen hat. Die Verfügbarkeit eines Computersystems wird definiert als das Verhältnis der tatsächlichen Betriebszeit zur geplanten Betriebszeit. Betriebsausfallzeiten entstehen aus den unterschiedlichsten Gründen, etwa durch Störungen in der Stromversorgung, Anwenderfehler, Hardware- oder Softwarefehler. Hohe Verfügbarkeit nimmt gerade in einem Mehrplatzsystem einen hohen Stellenwert ein, da bei einem Systemausfall eine größere Anzahl von Benutzern gleichzeitig betroffen ist. Hohe Verfügbarkeit wird erreicht, wenn sowohl Hardware- als auch Softwarekomponenten des Systems eine geringe Ausfallwahrscheinlichkeit aufweisen. Ein Ziel des Qualitätskonzepts von Siemens Nixdorf besteht darin, sowohl für Hardware als auch für Software eine geringe Ausfallwahrscheinlichkeit zu erreichen.

Neben den Qualitätsanforderungen an Hard- und Software kann die Verfügbarkeit durch zwei weitere Ansätze erhöht werden: Redundanz und Beschleunigung des Neustarts zur Wiederaufnahme des regulären Betriebs. Redundanz ermöglicht den Weiterbetrieb des Systems beim Ausfall einer Einzelkomponente. Ist ein Gesamtausfall jedoch unvermeidbar, muß der reguläre Betrieb möglichst schnell wiederhergestellt werden können, um den Schaden so gering wie möglich zu halten.

Das SINIX-Konzept der hohen Verfügbarkeit beinhaltet:

- Schutz vor Datenverlust und Inkonsistenzen
- erhöhte Betriebssicherheit
- Maßnahmen zur Vermeidung von Systemausfällen
- kontrollierte Systembeendigung im Falle eines Ausfalls
- verkürzte Wiederanlaufzeiten

Eine Stärke von SINIX ist es, ein ganzes Spektrum an Funktionen zur Erhöhung der Verfügbarkeit anzubieten. SINIX geht in dieser Hinsicht weit über die Möglichkeiten des Standard-UNIX hinaus. Durch diese Systemerweiterungen kann der Anwender die Verfügbarkeit des Gesamtsystems seinen Anforderungen entsprechend anpassen.

SINIX-Basisverfügbarkeit

Die im folgenden beschriebenen Funktionen sind Bestandteil jedes SINIX-Grundsystems.

Eindeutige Systemmeldungen

Die ersten Maßnahmen, um einen Komponenten- oder Geräteausfall zu verhindern, sind eindeutige Warnmeldungen, die die Identifizierung und Lokalisierung eines Problems erleichtern. Die SINIX Systemmeldungen der Gerätetreiber sind nach einem einheitlichen Schema aufgebaut. Eine eindeutige Meldungskennung erleichtert das schnelle Auffinden der zugehörigen Erklärung im Handbuch. Zusätzlich sind die Meldungen nach Gerätetypen klassifiziert. Jede Meldung ist mit einem Zeitstempel und einer Fehlergewichtung versehen, die bereits erste Hinweise auf die Schwere eines Fehlers und seine möglichen Auswirkungen gibt. Die Fehlergewichtungen sind so geordnet, daß für eine Schnelldiagnose bereits die für die Fehlerbehebung zuständige Instanz festgestellt werden kann (z.B. Operator, Teleservice, Wartung). Die Meldungen können sowohl auf der Konsole ausgegeben als auch in eine lokale Protokolldatei geschrieben werden. Im Netzverbund können sie zudem an einen zentralen Log-Server geleitet werden.

Verkürzung der Wiederanlaufzeiten

Ein möglicher Nebeneffekt der gepufferten Ein- und Ausgabe kann ein Datenverlust im Puffer sein, falls das System abrupt beendet wird, bevor der Puffer auf die Festplatte zurückgeschrieben werden konnte. Dadurch kann die Verwaltungsinformation über die Dateisysteme auf der Festplatte inkonsistent werden. Unter UNIX wird bei einem Neustart des Systems das Dienstprogramm *fsck* (file system check) ausgeführt, um alle Dateisysteme auf ihre Konsistenz zu prüfen und gegebenenfalls zu rekonstruieren. Je nach Größe des Dateisystems und der Anzahl der Festplatten kann diese Überprüfung mehr als zehn Minuten in Anspruch nehmen. Siemens Nixdorf hat *fsck* weiterentwickelt, um die Wiederanlaufzeit auf SINIX-Systemen dadurch zu verkürzen, daß Dateisysteme auf verschiedenen Festplatten parallel überprüft werden. Auf diese Weise kann die Überprüfung eines Dateisystems auf ein Drittel der Zeit verkürzt werden.

Systemstart von einer alternativen Festplatte aus

Ein einfacher, aber sehr wirkungsvoller Mechanismus bei einem Ausfall der Systemplatte besteht darin, eine alternative Festplatte bereitzuhalten, auf der die für einen Systemstart notwendige Software vorhanden ist. Falls die Systemplatte einen schwerwiegenden Fehler aufweist, der den Weiterbetrieb des Systems verhindert, kann das Betriebssystem jederzeit von der zweiten Systemplatte aus gestartet werden. Die Ausfallzeiten können so sehr kurz gehalten werden. SINIX stellt diese Funktionalität durch ein Kommando zur Verfügung, das den Systemverwalter bei der Einrichtung einer alternativen Systemplatte unterstützt. Dabei werden die relevanten Systemteile der aktuellen Konfiguration auf die neue Festplatte übertragen. Gleichzeitig werden alle geräteabhängigen Informationen automatisch an die neue Systemplatte angepaßt.

Optimierung der Hauptspeicherabzüge

Bei einem Systemausfall wird versucht, einen Abzug des Hauptspeichers vor der kompletten Beendigung auf die Festplatte zu schreiben (Systemdump). Bei größeren Hauptspeicherausbauten kann das Ablegen eines Speicherabzugs sehr zeitaufwendig sein und viel Festplattenplatz in Anspruch nehmen. Auf SINIX-Systemen wird der Hauptspeicher bei einem Systemabsturz deshalb in komprimierter Form auf der Festplatte abgelegt. Die kompakte Version macht nur einen Bruchteil der ursprünglichen Größe des Hauptspeichers aus. Sie wird zuerst im Swap-Bereich der Systemplatte abgelegt. Die geringe Größe des Hauptspeicherabzugs und seine schnelle Zwischenspeicherung garantieren, daß die Ausfallzeit des Systems kurz gehalten und das System sofort wieder neu gestartet werden kann. Erst während des folgenden Neustarts des Systems wird der Speicherabzug in das Dateisystem übernommen. Dort steht er für die spätere Diagnose zur Verfügung.

Höhere Verfügbarkeit unter SINIX

SINIX bietet eine Reihe zusätzlicher Produkte an, die die Systemverfügbarkeit für Installationen erhöhen, für die ein Höchstmaß an Verfügbarkeit unerläßlich ist. Die erhöhte Verfügbarkeit unter SINIX wird durch relativ preisgünstige und einfache Lösungen erreicht. Höhere Verfügbarkeit unter SINIX wird ohne exotische Hardware oder teure, hochspezialisierte Systeme erzielt.

Sichern des Gesamtsystems

Schutz vor Datenverlust kann durch die regelmäßige Sicherung der Systemplatte auf externe Datenträger erreicht werden. SINIX stellt hierfür das Programm Backup zur Verfügung, das dem Systemverwalter eine denkbar einfache Benutzerführung bietet. Es können vollständige Festplatten oder ausgewählte Dateisysteme gesichert werden. Folgende Bandmedien werden unterstützt: Magnetbandkassetten, Exabytekassetten oder Magnetbänder. Nach einem Ausfall der Systemplatte ist eine schnelle Wiederherstellung des Systems möglich, ohne daß eine Neuinstallation erforderlich wäre. Eine Gesamtsicherung der Systemplatte ist insbesondere nach der Installation neuer Software-Pakete empfehlenswert. Eine Beschreibung des Datensicherungsprogramms Backup finden Sie im Abschnitt „Backup" auf Seite 104.

Unterbrechungsfreie Stromversorgung

Um das Systemverhalten bei einer Unterbrechung der Netzspannung zu verbessern, kann das System mit einer unterbrechungsfreien Stromversorgung ausgestattet werden. Ein System, das mit einer unterbrechungsfreien Stromversorgung ausgestattet ist, stürzt nicht einfach ab, sobald ein Spannungsabfall registriert wird, sondern wird kontrolliert beendet. Die Daten werden gespeichert und das System wird vor dem Beenden in einen stabilen Zustand überführt. Dies erleichtert den Wiederanlauf und verkürzt die Ausfallzeit. Falls eine Netzstörung nur kurz andauert, kann die unterbrechungsfreie Stromversorgung die Beendigung des Systems sogar verhindern.

Eine eingebaute Batterie bietet noch höheren Schutz vor Unterbrechungen in der Stromversorgung. Diese Batterie überbrückt Netzstörungen bis zu einer Dauer von einigen Minuten. Im Fall längerer Störungen wird das System kontrolliert beendet. Datenverluste können so minimiert werden. Sobald der Strom wieder zur Verfügung steht, kann das System neu gestartet werden.

Spiegelplatten

Mit dem Virtuellen Plattensubsystem VPSS können Datenbestände ganzer Festplatten oder Teile davon (Slices) mehrfach gespeichert werden. Durch diese Redundanz wird garantiert, daß bei Ausfall eines Datenträgers die Daten von einer anderen Festplatte gelesen werden und Anwendungen ohne Störung weiterarbeiten können. VPSS spiegelt zusätzlich einzelne Slices einer Festplatte auf andere Slices oder andere Festplatten. Dabei werden alle Schreibvorgänge, die auf der primären Festplatte stattfinden, automatisch auch auf den sekundären Festplatten ausgeführt. So sind alle Festplattenslices, die zu einer Spiegelmenge gehören, immer auf einem einheitlichen Stand.

Sobald auf einer der Spiegelplatten oder dem Zugriffspfad ein Übertragungsfehler erkannt wird, werden die Datenzugriffe auf eine der redundanten Festplatten umgelenkt. Der Anwender ist durch den Fehler nicht betroffen. Das Spiegelverfahren kann sowohl für SINIX-Dateisysteme als auch für Datenbanken eingesetzt werden, die auf zeichenorientierten Slices (raw Slices) organisiert sind.

Umschaltbare Festplatten

Festplatten, die sich in einem eigenen, über SCSI angeschlossenen Peripherieschrank befinden, können abwechselnd von jeweils einer von zwei Anlagen angesprochen werden. Dies funktioniert über einen Schalter, der von der Software kontrolliert wird. Bei einem Systemausfall können die so verwalteten Datenbestände von einem zweiten System übernommen werden.

In Kombination mit dem Spiegelplattensystem VPSS können Doppelrechnerkonfigurationen betrieben werden, die ein hohes Maß an Ausfallsicherheit gewährleisten.

Standby-Konfigurationen

Noch höhere Verfügbarkeit kann mit Standby-Konfigurationen erzielt werden, die redundante Hardware ausnutzen.

OBSERVE

Ein zusätzlicher Schritt in einem System mit Standby-Konfiguration bildet die Installation von Software, die eine ständige Lebendüberwachung gewährleistet. Das Softwareprodukt OBSERVE von Siemens Nixdorf verfügt über diese Funktionalität. OBSERVE ist ein System zur Steuerung und Überwachung von Doppel- und Mehrrechnerkonfiguration. Die wichtigsten Peripheriegeräte werden über einen Schalter an beide Systeme angeschlossen, externe Festplatten sind umschaltbar angelegt.

OBSERVE automatisiert den Vorgang der Rechnerüberwachung durch eine ständige Lebendüberwachung der Rechner untereinander und übernimmt zusätzliche Überwachungsaufgaben. Falls einer der Rechner ausfällt, steuert OBSERVE die Übernahme seiner Aufgaben durch den Standby-Rechner. OBSERVE schaltet externe Festplatten, Terminals, Drucker usw. dem Standby-Rechner zu.

OBSERVE ermöglicht eine hohe Flexibilität in der Konfigurierung des Systems durch frei editierbare Konfigurationsdateien, editierbare Shellskripts zur Reaktion auf anwenderdefinierte Ereignisse sowie durch die Bereitstellung einer Schnittstelle (library) zur Erzeugung lokaler Überwachungsprogramme (Observer). So kann sich der Anwender die Verfügbarkeitsfunktionalität für seine Anforderungen maßschneidern. Abbildung 9-1 zeigt den Einsatz von OBSERVE in einer Standby-Konfiguration.

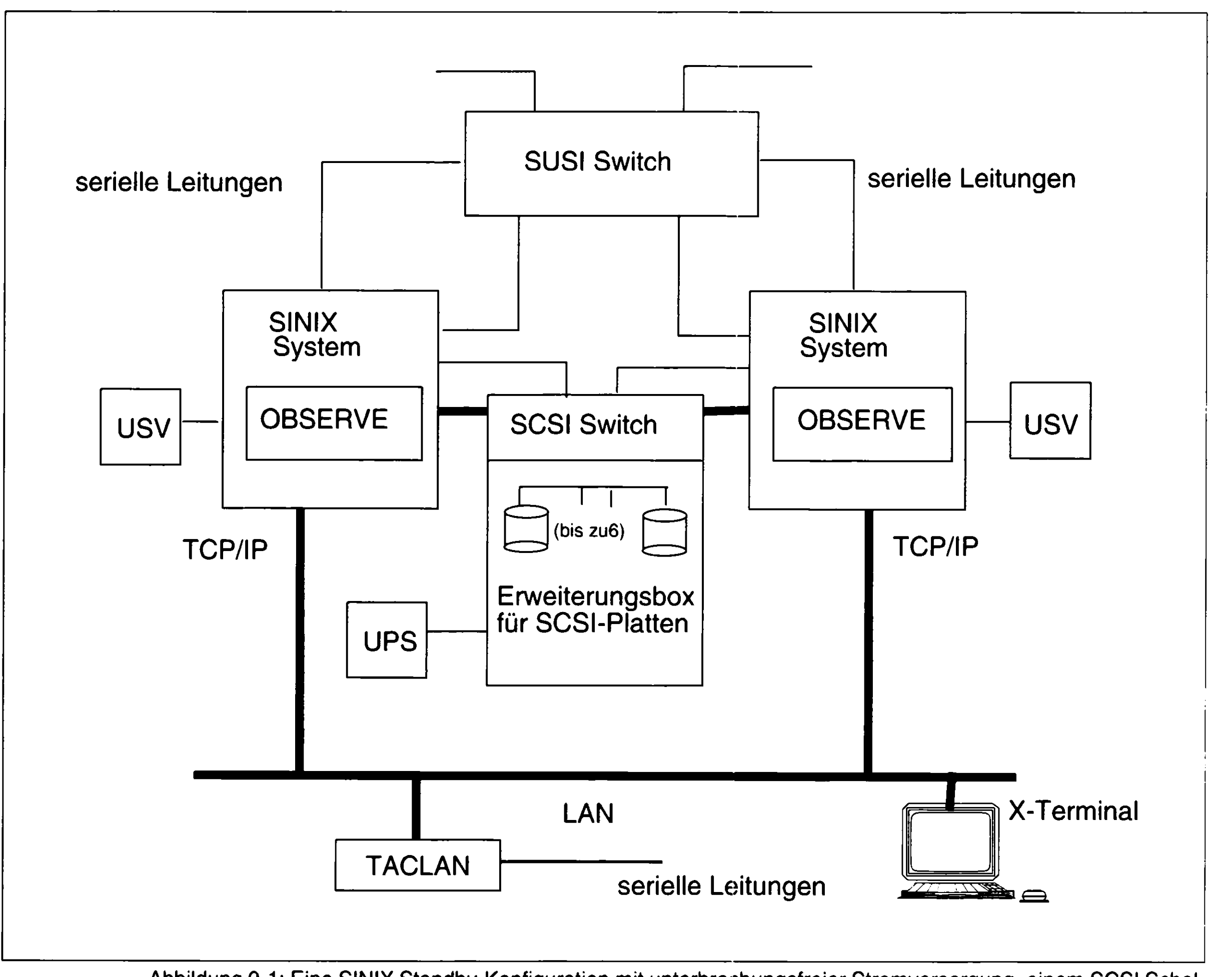


Abbildung 9-1: Eine SINIX Standby-Konfiguration mit unterbrechungsfreier Stromversorgung, einem SCSI Schalter, einem SUSI Schalter und dem Softwareprodukt OBSERVE.

Wesentliche Bestandteile eines Doppelrechnersystems sind

- die Extension Box für SCSI-Platten,
- serielle Umschalteinheiten, z.B. ein SUSI Schalter,
- der Terminal-Server TACLAN 91863;
- zwei Überwachungsleitungen zur gegenseitigen Rechnerüberwachung,
- wahlweise unterbrechungsfreie Stromversorgung an jedem Rechner.

Das Softwareprodukt OBSERVE besteht aus einer Reihe von Komponenten, wie in Abbildung 9-2 zu sehen ist. Das Hauptprogramm ist die zentrale Komponente und steuert den gesamten Ablauf von OBSERVE. Die Standardobserver PRT und EXIST sind Überwachungsprogramme, die mit OBSERVE ausgeliefert werden. Der Standardobserver PRT überwacht Drucker, während EXIST die Existenz von Prozessen überwacht. Eine Konfigurationsdatei enthält alle für die Überwachung notwendigen Daten über die Hardware- und Softwarekonfigurationen. Shellskripts, die vom Anwender erstellt werden, legen fest, wie auf Statusänderungen u.ä. reagiert werden soll.

Bei der Statusänderung von Objekten werden Objektreaktionsskripts ausgeführt. Bei Observerausfällen, die OBSERVE selbst erkennt, werden Observerreaktionsskripts ausgeführt. Für die Verwaltung von OBSERVE gibt es eine Reihe von Menüs und Kommandos, die entweder die Verwaltungsfunktionen selbst ausführen oder an das OBSERVE-Hauptprogramm weiterleiten. Ein Datenfernübertragungsmodul wickelt die Kommunikation zwischen den beiden Rechnern über Socketverbindungen über TCP/IP ab, die bei der Konfigurierung angegeben wurden. Dieses Modul überwacht die Leitungen und erkennt den Ausfall des Partners (Lebendüberwachung). Der Standardobserver ALIVE ermöglicht es, den Ablauf eines Prozesses (z.B. eines Anwendungsprogramms) zu überwachen. Dazu sind entsprechende Aufrufe der Bibliothek *libobsalive.a* in das Anwendungsprogramm zu integrieren.

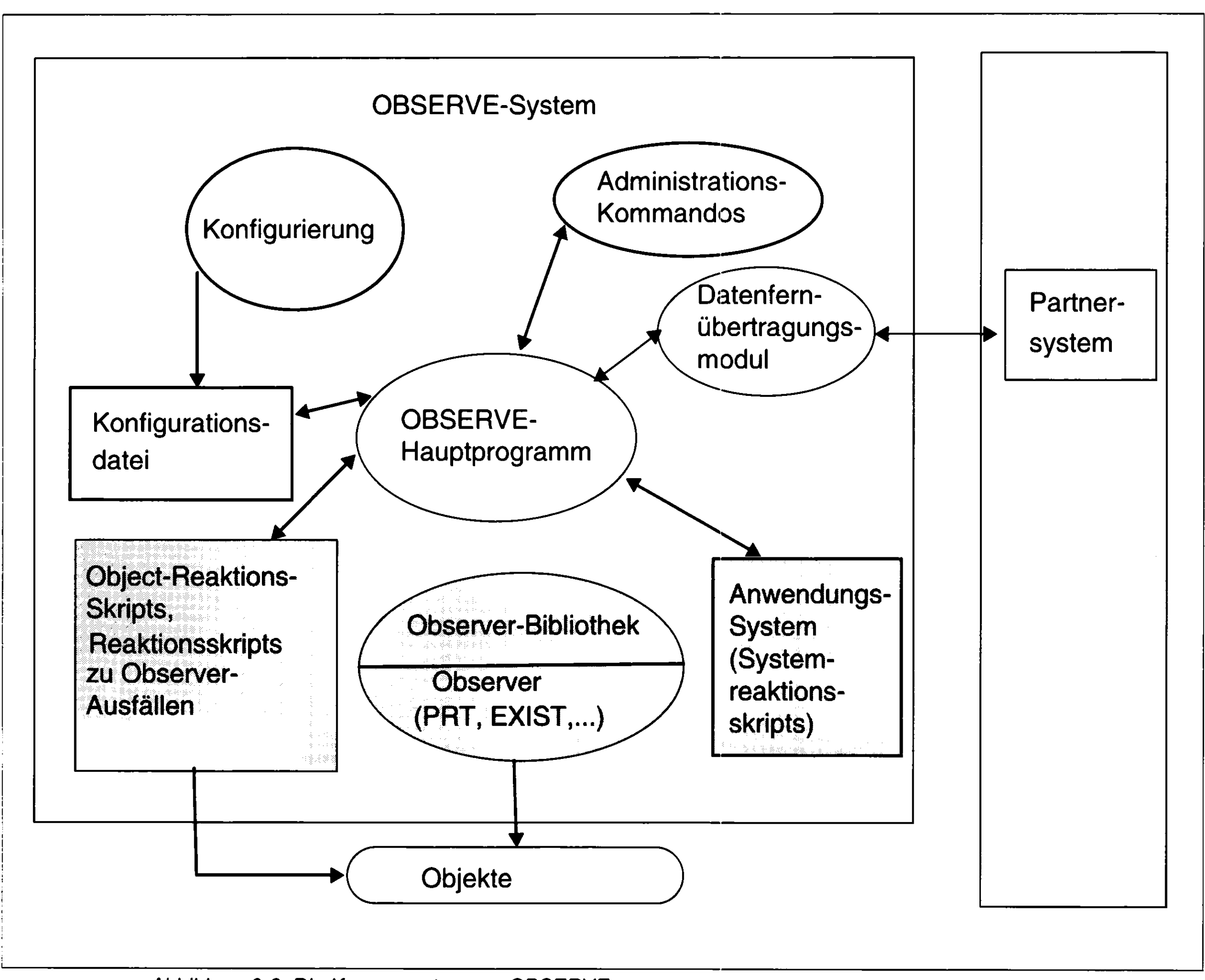


Abbildung 9-2: Die Komponenten von OBSERVE.

VxFS

Das VERITAS Filesystem (VxFS) ist ein Extent-basiertes Dateisystem für den Einsatz auf SINIX-Systemen. Das bedeutet, daß VxFS auf zusammenhängenden Speicherbereichen auf der Festplatte (Extents) basiert, im Unterschied zu den physikalisch verstreut liegenden Blöcken in herkömmlichen Dateisystemen. VxFS stellt Erweiterungen zur Verfügung, durch die SINIX in bestimmten kommerziellen Anwendungen noch besser einsetzbar ist. Die Haupteigenschaften des VxFS-Dateisystems sind:

- schnelle Wiederherstellung des Dateisystems
- Online-Verwaltung
- Online-Sicherung
- erweiterte Anwendungsschnittstelle
- Modus für die erweiterte Datenintegrität
- Verwendbarkeit als temporäres Dateisystem
- Verwendbarkeit als Root-Dateisystem

Das VxFS-Dateisystem ermöglicht die sekundenschnelle Wiederherstellung nach einem Systemausfall durch ein Fehlerverfolgungswerkzeug, das sogenannte Intent Logging. Das Intent Logging verfügt über eine zirkuläre Struktur, die noch nicht vollzogene Veränderungen in der Dateisystemstrukur ablegt, d.h. die Absicht einer Strukturänderung im Dateisystem wird aufgezeichnet. Diese Veränderungen werden in einer Protokolldatei aufgezeichnet. Bei der Wiederherstellung nach einem Systemausfall untersucht das VxFS-Kommando *fsck* (eine modifizierte Version des Standard-Kommandos *fsck*) diese Protokolldatei, um die Dateisystemoperationen, die zum Zeitpunkt des Systemabsturzes aktiv waren, aufzuheben oder zu vervollständigen. Das Dateisystem kann dann wieder in das Gesamtsystem eingehängt werden, ohne daß die strukturelle Überprüfung des gesamten Dateisystems notwendig ist. Abgesehen davon, daß die Wiederherstellung des VxFS-Dateisystems in wenigen Sekunden abgeschlossen ist, bleibt das Intent Logging für den Anwender oder den Systemverwalter i.a. unsichtbar. Das VxFS-Dateisystem verfügt zudem über die Eigenschaft, die Festplattenauslastung zu verbessern (Defragmentierung) und die Größe des Dateisystems anzupassen, während es in Benutzung ist und für den Anwender verfügbar bleibt.

Das VxFS-Dateisystem erlaubt eine konsistente Datensicherung während des Systembetriebs. Dabei macht VxFS von den Dateiverwaltungsstrukturen eines eingehängten Dateisystems eine Kopie. Dies geschieht in sehr kurzer Zeit, da die Daten nicht komplett kopiert werden. Die Daten in dieser Kopie bleiben „eingefroren", auch wenn zu einem späteren Zeitpunkt Änderungen am Dateisystem vorgenommen werden. Den aktuellen Stand mit diesen Änderungen nimmt das Original an. So können die die Originaldaten weiter bearbeitet werden, während die „eingefrorene" Kopie gesichert wird.

Im allgemeinen ist VxFS gegenüber Anwendungen transparent. Jede Anwendung, die auf einem bestehenden SINIX-Dateisystem ablauffähig ist, funktioniert mit VxFS identisch.

Ausblick

Die unterschiedlichen Ansätze, die bisher beschrieben wurden, werden auch in Zukunft dazu benutzt werden, eine höhere Verfügbarkeit zu erzielen. Verfügbarkeit gewinnt darüber hinaus für Systeme im Netzverbund durch verteilte Anwendungen, die auf Client-Server-Architekturen basieren, und den Einsatz mehrfach gehaltener Server an Bedeutung. In modernen Client-Server-Umgebungen wie derjenigen, die von DCE zur Verfügung gestellt wird, ist der Client einer Anwendung auf jedem System im Netzverbund verfügbar, das diese Anwendung einsetzt (siehe Abschnitt „Verteilte Verarbeitung: DCE als Lösungsmodell" auf Seite 132). Die Abhängigkeit von einem einzigen System wird so reduziert. Server werden im allgemeinen auf mehreren Rechnern im Netzverbund gehalten. Durch den Einsatz standortunabhängiger Namenskonventionen können Anfragen der Clients vom Server transparent umgeleitet werden. Dies wird notwendig, sobald ein Server nicht verfügbar ist, sei es, weil der Rechner, auf dem er sich befindet, nicht in Betrieb ist oder weil eine Netzstörung aufgetreten ist.

Sicherheit

Wenn große, einsatzkritische Anwendungen auf ein SINIX-System oder einen Netzverbund von SINIX-Systemen übertragen werden, stellen Benutzer, die an die Sicherheitstechniken in Mainframe-Systemen gewohnt sind, Fragen bezüglich der auf SINIX-Systemen verfügbaren Sicherheit. Der folgende Abschnitt gibt einen kurzen Überblick über die Systemsicherheit unter SINIX.

Da diese Anwendungen üblicherweise in einem Netzverbund ausgeführt werden, sind sie von der Sicherheit des Netzes abhängig. In der Fachpresse wird des öfteren der Eindruck erweckt, Netzwerke offener Systeme seien besonders unsicher. Dies ist grundsätzlich nicht der Fall. Allerdings sind offene Systeme im Netzwerk den Angriffen von Hackern ausgesetzt. Es ist jedoch unbestreitbar, daß Jahre der Erfahrung in der Abschirmung offener Systeme vor Hackern eine Reihe effektiver Sicherheitstechniken für Netzwerke hervorgebracht haben, sowie ein solides Wissen darüber, wie Netzwerke offener Systeme abgesichert werden können.

Für den Fall, daß bisher unbekannte Sicherheitsprobleme im Netzverbund offener Systeme auftreten, steht weltweit eine große Anzahl von Spezialisten bereit, um diese zu bekämpfen. Zusätzlich gibt es globale Netze wie das Internet und elektronische Bulletinboards, über die Sicherheitsinformationen Online verbreitet werden. Die Sicherheit im Netzverbund offener Systeme ist so einer ständigen weltweiten Überprüfung unterworfen, die zu entsprechenden Qualitätsverbesserungen führt. Insbesondere die Veröffentlichung möglicher Sicherheitsprobleme führt zu deren Beseitigung.

Über ein Netzwerk verbundene Informationssysteme haben bisher unerreichte Möglichkeiten der Informationsverarbeitung und Kommunikation eröffnet. Dies führte jedoch auch zur Abhängigkeit von der Sicherheit und der Verfügbarkeit dieser Systeme. Mit dem Durchbruch UNIX-basierter Systeme auf dem kommerziellen Markt wurden auch entsprechende Sicherheitsanforderungen an deren Entwicklung gestellt und realisiert. Gleichzeitig wurden die Ergebnisse anderer Gruppen, die sich mit der Sicherheit der Informationstechnologie (IT-Sicherheit) befaßten, in den ständigen Entwicklungsprozeß UNIX-basierter Systeme wie etwa SINIX miteinbezogen.

Aus den Evaluationskriterien für sichere IT-Systeme in den USA und Europa ergaben sich wichtige technische Anforderungen. Der verbreitete Einsatz von UNIX-Systemen im Netzverbund erfordert die Integration von Sicherheitsstandards und De-facto-Standards in TCP/IP- und OSI-Netzwerken. Auch auf Workstations, die auf dem X Window System basieren, muß ein sicherer Betrieb möglich sein. Nicht zuletzt müssen auch für systemnahe Softwareprodukte wie z.B. Datenbanksysteme zusätzliche Sicherheitsfunktionen entwickelt werden.

Basissicherheit unter SINIX

Ein SINIX-System in der Standard-Konfiguration beinhaltet bereits erhebliche Sicherheitsfunktionen. Das SINIX V5.4x Basissystem beispielsweise erfüllt die Anforderungen für die Sicherheitsklasse C1, wie sie im Anforderungskatalog der U.S. Regierung TCSEC (**T**rusted **C**omputer **S**ecurity **E**valuation **C**riteria, auch unter dem Namen „Orange Book" bekannt) formuliert wurden. Siemens Nixdorf testet die Effektivität dieser Funktionen regelmäßig im Rahmen der Qualitätssicherung des Betriebssystems. Die Mechanismen zur SINIX-Systemsicherheit umfassen:

Benutzeridentifikation und Authentisierung

Jeder Benutzer muß dem System bekannt sein (Identifikation) und bei der Anmeldung am System durch die Eingabe eines nur ihm bekannten Paßwortes seine Identität nachweisen (Authentisierung).

Benutzergruppen

Zusätzlich zur Identifikation einzelner Benutzer ermöglicht das System die Bildung von Gruppen. Mehrere Benutzer können so zu einer Gruppe zusammengefaßt werden, die auf bestimmte Daten gemeinsamen Zugriff hat. Dies ist sinnvoll für Unternehmensabteilungen wie beispielsweise eine Personalabteilung, deren Mitarbeiter auf eine gemeinsame Datenbank zugreifen müssen, die für andere Angehörige des Unternehmens jedoch nicht zugänglich sein soll.

Zugriffsberechtigungen

Jedem Datenobjekt, d.h. Dateien, Dateiverzeichnissen und den sogenannten Objekten der Interprozeßkommunikation wie Shared Memory, Semaphoren und Message Queues, werden bestimmte Attribute, die sogenannten Permissionbits, zugeordnet.

Zugriffsberechtigungen

Jedem Datenobjekt, d.h. Dateien, Dateiverzeichnissen und den sogenannten Objekten der Interprozeßkommunikation wie Shared Memory, Semaphoren und Message Queues, werden bestimmte Attribute, die sogenannten Permissionbits, zugeordnet. Permissionbits regeln den Zugriff auf das jeweilige Datenobjekt. Folgende drei Zugriffsarten werden unterschieden: lesender, schreibender und ausführender Zugriff. Bei der Entscheidung, wer auf ein Objekt zugreifen darf, teilt das System alle Benutzer in drei Klassen ein: den Eigentümer, die Gruppe und die übrigen Benutzer. Für jede dieser Benutzerklassen können die Zugriffsberechtigungen auf ein Objekt getrennt festgelegt werden (Lese-, Schreib- und Ausführberechtigung).

Privilegierte Benutzer

Sicherheitsrelevante Systemveränderungen wie etwa die Verwaltung von Zugriffsrechten oder die Einrichtung neuer Benutzer können nur von einem privilegierten Benutzer, dem Systemverwalter, durchgeführt werden.

Prozesse

Jeder Prozeß im System besitzt einen eigenen Adreßraum, der von den Adreßräumen anderer Prozesse abgegrenzt ist. Die Kommunikation zwischen unabhängigen Prozessen ist nur über spezielle Mechanismen möglich, die wiederum durch die Regelung der Zugriffsrechte überwacht werden.

Erweiterte Sicherheitsfunktionen unter SINIX

SINIX-Versionen mit erweiterter Sicherheit bieten wichtige zusätzliche Sicherheitsfunktionen wie Zugriffskontrolle, Rollenkonzept und Protokollierung. Im Rahmen von DCE werden zusätzliche Sicherheitsfunktionen für verteilte Systeme in SINIX integriert. SINIX-Systeme werden dadurch mit mächtigen zusätzlichen Netzwerk-Sicherheitsfunktionen wie Authentisierung, Authorisierungsprüfung und Datenverschlüsselung ausgestattet (siehe Abschnitt „DCE-Komponenten" auf Seite 135). Zusätzlich wird die Zugangskontrolle in SINIX-Systeme durch Chipkarten unterstützt. In Zukunft wird der Arbeitsschwerpunkt auf erweiterten Sicherheitsfunktionen (SINIX-ES, basierend auf UNIX SVR4-ES der UNIX System Laboratories,) und allgemeinen Sicherheits-APIs (Application **P**rogramming **I**nterfaces) liegen. Parallel dazu verfolgt Siemens Nixdorf die Kompatibilität und Interoperabilität von SINIX in offenen Systemen durch die aktive Mitarbeit an der Entwicklung von Sicherheitsstandards.

Neben der Sicherheit im Basissystem existieren zwei Ansätze zur weiteren Verbesserung der Sicherheitseigenschaften von SINIX:

- Weiterentwicklung der Sicherheitsfunktionen zu einem System, das einer Evaluationsinstanz zur Zertifizierung vorgelegt werden kann.
- Bereitstellung zusätzlicher Funktionen, die die Basissicherheit erhöhen.

Eine Beschreibung evaluierter Produkte finden Sie im Abschnitt „Evaluierte Sicherheit unter SINIX" auf Seite 197. Der folgende Abschnitt beschäftigt sich mit zwei Aufsatzprodukten, die zusätzlich zur SINIX-Basisversion installiert werden können und jeweils zusätzliche Sicherheitsfunktionalität zur Verfügung stellen.

ASECO

Der Schwachpunkt bei der Sicherheit von IT-Systemen ist die Authentisierung. Authentisierung eines Benutzers bedeutet, daß dieser während der Anmeldung am System seine Identität nachweist. Dieser Nachweis wird in herkömmlichen Systemen meist mittels eines Paßworts erbracht, von dem man annimmt, daß es nicht leicht zu erraten und nur dem Benutzer bekannt ist. Die Praxis zeigt jedoch, daß Paßwörter nicht sicher sind:

- Paßwörter sind häufig leicht zu erraten, wie etwa der Name des Ehepartners, der Name des Benutzers, der Rechnername oder einfache Variationen dieser Namen, z.B. durch das Anfügen einer Ziffer.
- Das Paßwort wird aufgeschrieben und die Notiz offen liegen gelassen, beispielsweise in der Nähe eines Terminals.
- Es sind Programme verfügbar, auch über Public Domain, die ausgefeilte Aktionen zum intelligenten Erraten von Paßwörtern durchführen. Diese Programme greifen auf ein elektronisch zugängliches Wörterbuch zurück, dessen Inhalt variiert werden kann.
- Da offene Systeme häufig im Netzverbund verwendet werden, findet die Anmeldung am System meist über das Netzwerk statt. In diesem Fall wird das Paßwort möglicherweise unverschlüsselt über das Netz übertragen und damit für andere leicht lesbar und späterem Mißbrauch ausgesetzt.
- Ein häufiger, besonders schwerwiegender Fall liegt bei der Anwendung von Netzkommandos wie *remote copy* vor. Diese verlangen, daß die Paßwortabfrage im Zielrechner unterdrückt wird und lediglich durch Prüfung des lokalen Rechners und des Namens des Benutzers, der das ferne Kommando ausführen will, ersetzt wird. Dieser Mechanismus bewirkt zugleich, daß die Anmeldung am fernen Rechner ebenfalls ohne Paßwortabfrage erfolgt.

ASECO unterbindet all diese Schwachpunkte: der Zugang zu SINIX-Systemen wird von ASECO mit Hilfe einer Chipkarte geregelt, die kryptographische Verschlüsselungen unterstützt.

Ein Benutzer, der sich an einem durch ASECO geschützten SINIX-System anmelden will, muß im Besitz einer Chipkarte sein. Zusätzlich muß er die nur dieser Chipkarte bekannte PIN (**P**ersonal **I**dentification **N**umber) kennen, um die Anmeldung erfolgreich durchzuführen.

Die auf der Chipkarte gespeicherte geheime Information ist nicht lesbar. Dies gilt insbesondere für die kryptographischen Schlüssel. Die Chipkarte benutzt einen kryptographischen Algorithmus für diese geheimen Schlüssel. Das bedeutet, daß es keinerlei Möglichkeit gibt, den geheimen Schlüssel der Chipkarte zu lesen. Die der Chipkarte bekannte PIN ist ebenfalls nicht lesbar, sie kann jedoch geändert werden, um die Gefahr eines möglichen Mißbrauchs der Chipkarte auszuschließen.

Authentisierung ist ihrem Prinzip nach zweiseitig. Eine Instanz erbringt gegenüber einer zweiten Instanz den Nachweis, daß sie über die von ihr vorgegebene Identität verfügt. Da es sich bei der einen Instanz um die Chipkarte handelt, muß die zweite Instanz - der Rechner - ebenfalls über denselben geheimen Schlüssel verfügen. Ein spezielles Sicherheitsmodul enthält diesen Schlüssel inklusive des Verschlüsselungsalgorithmus.

Prinzipiell findet die Authentisierung des Chipkartenbenutzers in zwei Schritten statt:

1. Nach Einlegen der Chipkarte weist sich der Benutzer durch Eingabe seiner PIN gegenüber der Chipkarte aus.

2. Anschließend wird der eigentliche Authentisierungsmechanismus zwischen Chipkarte und Sicherheitsmodul des Rechners durchgeführt. Die Authentisierung wird seitens der Chipkarte nur ausgeführt, wenn die vorangegangene PIN-Prüfung erfolgreich verlaufen ist.

Der Authentisierungsmechanismus beruht darauf, daß Zufallszahlen verschickt werden, die vom jeweiligen Partner verschlüsselt quittiert werden müssen (Challenge-Response-Verfahren).

Ein Mißbrauch der Chipkarte wird dadurch ausgeschlossen, daß der Authentisierungsmechanismus erst gestartet wird, nachdem die Chipkarte die Identität des Benutzers mittels der PIN festgestellt hat.

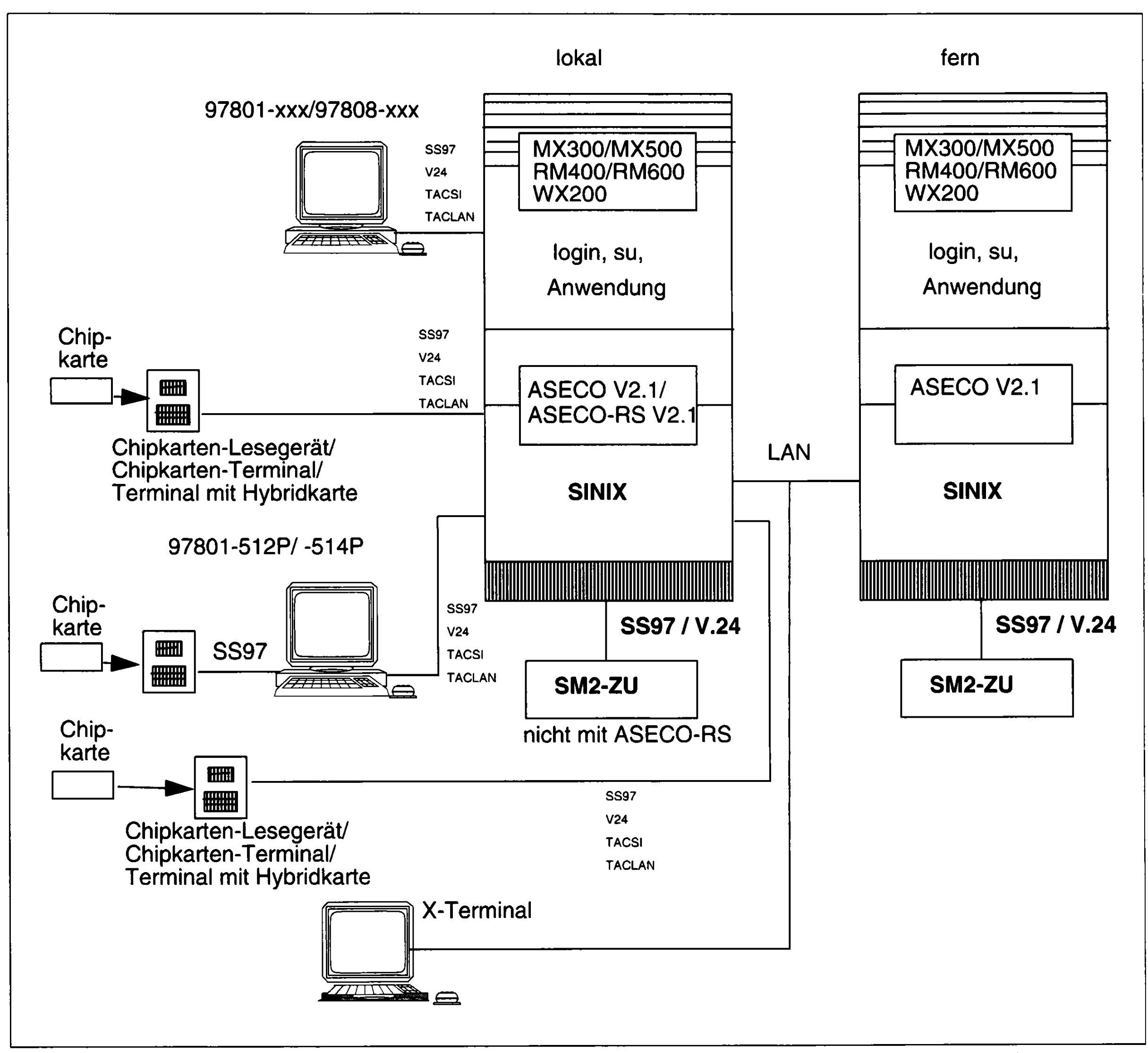


Abbildung 9-3: ASECO stellt für lokale und ferne SINIX-Systeme eine Sicherheitsschnittstelle für Chipkartenbenutzer zur Verfügung.

Weitere Leistungen von ASECO:

- Der von ASECO verwendetet Mechanismus zur Authentisierung steht auch als Programmierschnittstelle (API) zur Verfügung und kann so für kundeneigene Anwendungen eingesetzt werden. Der Zugang zu diesen Anwendungen kann dann mit Hilfe der Chipkarte kontrolliert werden.
- Ein weiters API gibt es für die Chipkartenüberwachung. Diese ermöglicht die Suspendierung einer Sitzung, sobald die Chipkarte aus dem Chipkartenterminal entfernt wird. Eine Fortsetzung der Sitzung ist erst möglich, wenn die Chipkarte erneut eingelegt und der Identifizierungs- und Authentisierungsvorgang erfolgt ist.
- ASECO und auch alle Anwendungen, die das ASECO-API verwenden, sind auch im Netzverbund funktionsfähig; Chipkarten-Terminals und Benutzer-Terminals können auch an ferne Rechner im Netz angeschlossen werden. Dies ist von großer Bedeutung, da so ein Angriff verhindert wird, der durch Abhören eines im Netz übertragenen Paßwortes leicht möglich gewesen wäre.
- Sowohl die Verbindung zwischen Sicherheitsmodul und System als auch die Verbindung zwischen System und Chipkarte kann ohne Verletzung der Sicherheit „abgehört“ werden. Im Authentisierungsverfahren werden lediglich Zufallszahlen und verschlüsselte Zufallszahlen über die Verbindung geschickt.
- Der Anschluß der Chipkarte an das System erfolgt über ein Chipkarten-Terminal, das neben dem physikalischen Anschluß der Chipkarte auch eine Tastatur besitzt, über die die PIN eingegeben wird. Die PIN wird dann an die Chipkarte zur Überprüfung weitergeleitet. Das Chipkarten-Terminal ist so konstruiert, daß die Verbindung zwischen Terminal und Chipkarte nicht abgehört werden kann. Die PIN ist ein Geheimnis zwischen der Chipkarte und ihrem rechtmäßigen Benutzer. Die PIN kann vom Benutzer über das Chipkartenterminal geändert werden.

Audit

Systeme, die der Sicherheitsklasse C2 des Orange Book entsprechen, bieten weitaus höhere Sicherheit als herkömmliche Systeme. Gleichzeitig halten sich der finanzielle Mehraufwand sowie der Mehraufwand für höhere Sicherheitsfunktionen, der mit der Verwaltung und Anpassung von Aufsatzprodukten verbunden ist, in Grenzen. Bei näherer Betrachtung eines auf UNIX SVR4-basierenden Systems wie SINIX muß festgestellt werden, daß die wichtigste fehlende Funktionalität, um C2 zu genügen, eine Komponente zur Protokollierung ist. Das Produkt Audit bietet diese Funktionalität. Mit Hilfe von Audit können sicherheitsrelevante Ereignisse, sogenannte Events, protokolliert werden. So kann der Systemverwalter später nachvollziehen, was sich zu einem beliebigen Zeitpunkt im System ereignete.

Beispiele für Events, die in SINIX protokolliert werden:

add_usr	neue Benutzerkennung einrichten
date	Systemdatum ändern
open_wr	Datei zum Schreiben öffnen
dac_mode	Zugriffsrechte einer Datei ändern
login	Anmeldung eines Benutzers.

Events werden im Kern des Betriebssystems beim Ablauf bestimmter Systemaufrufe (z.B., *date*, *open*, *chmode*) oder vertrauensrelevanter Kommandos (z.B., *add_usr*, *login*) erzeugt. So können sicherheitsrelevante Events (z.B. Zugangsversuche) erkannt und überwacht (protokolliert) werden.

Die Schnittstelle, die von den Kommandos verwendet wird (*auditdmp*), ist beschrieben und kann auch von vertrauenswürdigen Anwendungen verwendet werden. Die Vertrauenswürdigkeit einer Anwendung wird dadurch sichergestellt, daß nur der Systemverwalter die entsprechende Berechtigung vergeben kann. Kriterien, die über eine Protokollierung von Events entscheiden, werden ebenfalls vom Systemverwalter festgelegt. Dabei hat er die Möglichkeit, die Kriterien systemweit für alle Benutzer oder aber speziell für bestimmte Benutzer festzulegen.

Die Daten werden in einem kompakten Format auf ein Speichermedium geschrieben. Zur Auswertung steht ein spezielles Kommando (*auditrpt*) zur Verfügung, das die Daten in ein lesbares Format umwandelt und einfache Selektionen ermöglicht (z.B. alle Events für einen Benutzer). Als Speichermedien werden Dateien benutzt. Zur Verwaltung des Audit-Systems (z.B. Ein- und Ausschalten) steht eine Reihe von Kommandos zur Verfügung. Die Aktionen können jedoch auch menügesteuert über das Bediensystem zur Systemverwaltung ausgeführt werden.

Evaluierte Sicherheit unter SINIX

Bereits in den 60er Jahren begannen in den USA vorbereitende Arbeiten zur Entwicklung sicherer IT-Systeme und der zu diesem Zweck notwendigen Sicherheitsfunktionen. Diese Arbeiten führten 1983 schließlich zur Veröffentlichung eines Kriterienkatalogs, den „**T**rusted **C**omputer **S**ystem **E**valuation **C**riteria" (TCSEC, Orange Book). 1985 wurde eine aktualisierte Version herausgegeben. Für das Orange Book verantwortlich zeichnet das **N**ational **C**omputer **S**ecurity **C**enter (NCSC). Dort werden auch Evaluationen durchgeführt.

In Deutschland setzten entsprechende Aktivitäten 1989 mit der Gründung der **Z**entralstelle für **S**icherheit in der **I**nformationstechnik (ZSI) ein. Im gleichen Jahr veröffentlichte auch die ZSI Sicherheitskriterien, das sogenannte Grünbuch. Seit 1991 ist das **B**undesamt für **S**icherheit in der **I**nformationstechnik (BSI), das aus der ZSI hervorging, für die Evaluation und Zertifizierung sicherer IT-Systeme zuständig. Ebenfalls im Jahr 1991 wurde ein entsprechendes Evaluationshandbuch veröffentlicht. SINIX-S V5.22 wurde als erstes UNIX-System in Deutschland evaluiert und zertifiziert.

1990 veröffentlichten Frankreich, die Niederlande, Großbritannien und Deutschland erstmals ihre gemeinsamen IT-Sicherheitskriterien (ITSEC). Diese befinden sich als Version 1.2 in einer zweijährigen Erprobungsphase. Es ist zu erwarten, daß diese Kriterien europaweit Einsatz finden werden. Die Evaluierungs- und Zertifikationsverantwortung liegt bei den nationalen Behörden. Inzwischen wurden auch privatwirtschaftlich organisierte Unternehmen zugelassen, um Sicherheitsevaluationen durchzuführen.

Während sich TCSEC und ITSEC strukturell stark unterscheiden, sind die zu untersuchenden Sicherheitsfunktionen sehr ähnlich. Nach TCSEC sind die Funktionen hierarchisch von D als schwächster Gruppe über C1, C2, B1, B2, B3 bis zu A als stärkster Gruppe geordnet. In Gruppe C werden speziell Protokollierungsfunktionen, Zugangskontrolle und benutzerdefinierte Zugriffskontrolle untersucht. In Gruppe B kommen systemdefinierte Zugriffskontrolle und Rollendefinition hinzu. Die strukturellen Unterschiede zwischen ITSEC und TCSEC liegen darin, daß:

- im ITSEC qualitative Eigenschaften bezüglich der Wirksamkeit der Mechanismen und der Qualität ihrer Realisierung in eigenen Klassen/Kriterien festgelegt sind und
- im ITSEC die Funktionen so allgemein gefaßt wurden, daß auch Sicherheit in Datenbanken und in der Kommunikation sowie Zuverlässigkeitseigenschaften abgedeckt werden.

Weitere Informationen über Sicherheitsstandards finden Sie in Anhang A dieses Buches. Zusätzlich zu den bereits genannten Funktionen des Basissystems wird speziell im Rahmen offener Systeme an sicheren, X-basierten Workstations und sicheren verteilten Systemen gearbeitet. Darüber hinaus haben in den USA gemeinsame Arbeiten des NIST (**N**ational **I**nstitute for **S**tandards and **T**echnology) und der NSA (**N**ational **S**ecurity **A**gency) mit der Zielsetzung begonnen, TCSEC (das Orange Book) durch einen neuen Standard zu ersetzen. Ein erster Entwurf der Federal Criteria wurde bereits veröffentlicht.

SINIX-S

Aufsetzend auf bestehende Sicherheitsmechanismen des SINIX-Basissystems bietet SINIX-S Funktionen, die den C2-Anforderungen des Orange Book entsprechen. Das System wurde beim BSI einer dem Grünbuch entsprechenden Evaluation unterzogen und erzielte eine Einstufung in den Klassen F1/Q2. Der Evaluationsbericht ist über alle Geschäftsstellen von Siemens Nixdorf zu beziehen. Da Zugriffskontrollisten nach ITSEC zwingend vorgeschrieben sind, konnten die Anforderungen der Funktionsklasse F2 nicht vollständig erfüllt werden. Wesentliche funktionale Erweiterungen gegenüber dem SINIX-Basissystem bestehen u.a. in:

- der Einführung eines Audittrails zur Protokollierung und späteren Nachvollziehbarkeit sicherheitsrelevanter Operationen sowie
- der Einführung eines Rollenkonzepts (Vergabe von Privilegien) zur Trennung von Systemverwalter, Verwalter von Sicherheitsbelangen und Operator.

Mit dem XPG3plus-Zertifikat ist SINIX das einzige UNIX-System, das auch ein offizielles Sicherheitszertifikat entsprechend Q2 (Grünbuch) trägt.

SINIX-ES

SINIX-ES ist das Ergebnis einer Referenzportierung von UNIX SVR4.ES (Enhanced Security) auf Intel 386/486 Plattformen. UNIX SVR4.ES bietet Sicherheitsfunktionen entsprechend der Klasse B2 des Orange Book und wird derzeit vom NCSC evaluiert.

Folgende wesentliche Eigenschaften unterscheiden SINIX-ES von C2- oder B1-Systemen:

- Sicherer Systemzugang sichert den Vorgang der Anmeldung am System gegen Scheinprogramme, die eine Anmeldung vorspiegeln und sich dadurch Paßwörter erschleichen.
- Privilegien ermöglichen eine Trennung von Rollen im System. Systemverwalter, Verwalter von Sicherheitsbelangen und Operator sind vordefiniert, weitere Rollen sind jedoch konfigurierbar. Alle Systemprogramme unterstützen das Prinzip der minimalsten Privilegien, das die jeweils gültigen Privilegien auf das kleinste notwendige Maß beschränkt.
- Jeder Systemressource wird eine Sicherheitsklasse zur Unterstützung der systemdefinierten Zugangskontrolle zugewiesen.
- Der Kern wurde überarbeitet, um durch eine klare Modulstruktur die Evaluationsanforderungen erfüllen zu können. Die Anzahl globaler Variablen wurde reduziert. Der Kern wurde überprüft, um sicherzustellen, daß die Software keine „Hintertüren" enthält, die eine nicht vorgesehene Nutzung seiner Funktionalität ermöglichen.

Die unterstützte Sicherheitspolitik ist formal beschrieben. Ausführliche Designdokumentation steht zur Verfügung. Um die Sicherheitsfunktionen zu testen, wurde eine spezielle Sicherheitstestfolge entwickelt.

Qualitätsstandards

Das Betriebssystem SINIX konnte bisher die ständig wachsenden Anforderungen der Kunden an die Qualität erfüllen und wird dies auch weiterhin tun. Ebenso erwies sich SINIX der zunehmenden Komplexität der heute üblichen Softwarekonfigurationen gewachsen. Schon sehr bald machte unser Erfolg in dieser Hinsicht SINIX zu einem ausgereiften Betriebssystem für den kommerziellen Einsatz. Dies bedeutete die Abkehr von der herkömmlichen Einschätzung, UNIX-basierte Systeme seien als Betriebssysteme nur für Experten oder technische Anwender geeignet. Die „Kommerzialisierung" von SINIX wurde auf mehreren Wegen erreicht, u.a. durch die Übernahme von Prozeßschritten des Software-Engineering, die sich bereits im Bereich der Mainframe-Systeme bewährt hatten. Gleichzeitig wurden die Aufwände für die Qualitätssicherung auf über 30% der Gesamtentwicklungskosten erhöht. Die positiven Ergebnisse dieser Maßnahmen zeigen sich an geringeren Kosten für Wartung und Gewährleistung, die Siemens Nixdorf für SINIX vorweisen kann, trotz einer wachsenden Palette von SINIX-Produkten und steigender Marktdurchdringung.

Dem Druck des Wettbewerbs zur Unterstützung neuer Hardware und zusätzlicher Softwarefunktionalität in einem beschleunigten Innovationszyklus kann nur durch eine sehr hohe Entwicklungsproduktivität, eine große Flexibilität der Mitarbeiter und ein effektives Projektmanagement begegnet werden, das auf einem technologisch ausgereiften Entwicklungsprozeß basiert.

Siemens Nixdorf hält im Wettstreit zwischen „Time to Market" und Qualität letztendlich jedoch am Leitspruch „Qualität vor Termin vor Funktion" fest. Darunter verstehen wir, daß Termintreue wichtiger als zusätzliche Funktionalität ist, Qualität jedoch stets oberste Priorität hat, selbst wenn dies bedeuten sollte, daß Zeitpläne nicht genau eingehalten werden können. Keinesfalls sollen aufgrund von Termindruck Produkte ausgeliefert werden, die unsere anspruchsvollen Qualitätsstandards nicht erfüllen. Dieses Konzept ist in die internen Verträge zwischen Produktplanung und Entwicklung eingeflossen. Es wird außerdem durch die organisatorische Trennung der Bereiche Qualitätssicherung und Entwicklung unterstützt.

Die Testmethodik, die vor der Freigabe einer SINIX-Version Anwendung findet, erhöht sukzessive die Komplexität der getesteten Software- und Hardware-Umgebung, bis eine komplexe Software-Umgebung und eine komplexe Hardware-Konfiguration erreicht sind, in denen das Zusammenwirken von Datenbankanwendungen, Compilern, Netzwerkprogrammen und weiteren Produkten sichergestellt ist. So beschränkt sich die herausragende Qualität von SINIX-Systemen nicht nur auf das Betriebssystem selbst, sondern umfaßt auch ein breites Spektrum integrierter Produkte.

Wesentliche Merkmale guter Qualität sind:

- die hohe Verfügbarkeit der Software; Behebung möglichst aller Fehler vor der Kundenfreigabe (weniger als 2% erwarteter Fehler nach der Kundenfreigabe)
- einheitliche Bedienoberflächen von Low-End- bis zu High-End-Systemen
- hochwertige Dokumentation
- die Möglichkeit einer Ferndiagnose bei eventuell auftretenden Problemen sowie schnelle Bereitstellung von Korrekturen
- die Fähigkeit, Kundenanforderungen umzusetzten und die regelmäßige Bereitstellung funktionaler Upgrades
- überzeugende Leistung auch bei hohen Anforderungen an die Ressourcen
- ein gutes Preis-/Leistungsverhältnis.

Entwicklungsprozeß

Der Entwicklungsprozeß für SINIX-Systeme wird durch das Siemens Nixdorf **M**ethoden**h**and**b**uch (MHB) geregelt. Das MHB definiert eine Entwicklungsmethodik, die innerhalb von Siemens Nixdorf für die gesamte Entwicklung von BS2000- und SINIX-Systemen Anwendung findet. Diese Methodik ist in mehr als 15 Jahren herangereift und hat sich in diesem Zeitraum auch bewährt. Die Festlegungen des MHB werden regelmäßig überprüft und aktualisiert, um den Fortschritten in der Software-Entwicklungsmethodik gerecht zu werden. Zusammen mit den Regelungen des Qualitäts-Management-Handbuchs von Siemens Nixdorf entsprechen sie der international anerkannten Qualitätsnorm ISO 9001.

Der chronologische Prozeß bei der Entwicklung eines Produkts beginnt mit dem Zeitpunkt des innerbetrieblichen Vertragsabschlusses zwischen der Systemplanung und der Entwicklung. In diesem Vertrag wird das Produkt, das entwickelt werden soll, so genau wie möglich definiert, Kundenanforderungen und sonstige Entwicklungsziele werden festgehalten. Der Entwicklungsprozeß wird gemäß dem MHB durch die Definition von Meilensteinen in eine zeitliche Folge von Prozeßschritten zerlegt, ebenso werden die entsprechenden Verantwortlichkeiten geregelt. Die Ergebnisse jedes Schrittes werden durch verschiedene Verfahren verifiziert (Fagan-Reviews, Development Document Review). Protokolle jedes Verifizierungsverfahrens werden über den entsprechenden Standard erstellt und archiviert.

Bereits zum Zeitpunkt der Planung wird das Produkt, das entwickelt werden soll, in einzelne Teilaufgaben strukturiert. Diese werden entweder in parallelen oder genau aufeinander abgestimmten Entwicklungsteilschritten bearbeitet. Gleichzeitig wird ein erster Terminplan und ein Prognosemodell über die Fehler, die während der Testphase gefunden werden sollen sowie über die zu erwartende Leistung komplexerer Produkte erstellt. Falls ein Produkt auf Fremdcode basiert, wird die Qualität dieses Fremdcodes möglichst schon im Vorfeld der eigentlichen Entwicklung verifiziert.

Regelmäßig stattfindende Produkt-Status-Meetings kontrollieren den Projektfortschritt bei möglichst genauer Einhaltung des Produktplans. Notwendige Entscheidungen werden von einer genau festgelegten Gruppe innerhalb des Managements getroffen. Elektronische Post (E-Mail) und andere Methoden werden angewandt, um offene und effiziente Informationskanäle bereitzustellen.

Die CASE-Verfahren, die bei der SINIX-Entwicklung eingesetzt werden, gewährleisten eine gleichbleibend hohe Produktivität. Sie sind standortunabhängig und erfüllen die Voraussetzung für die dezentrale Entwicklung eines gemeinsamen Produkts. Die derzeitigen Entwicklungsstellen von Siemens Nixdorf sind München, Paderborn, Namur (Belgien), Wien (Österreich) und Burlington (USA).

Es werden leistungsfähige innerbetriebliche Werkzeuge verwendet, darunter:

SMS

Ein Sourcecode-Verwaltungssystem, das die Arbeit an demselben Code von verschiedenen Standorten aus erlaubt und Common-Source-Strukturen unterstützt. SMS wird benutzt, um mehr als 100.000 Dateien mit einem Speicherbedarf von über 1,5 GigaByte zu verwalten.

Local View Tools

Mit Hilfe dieser Werkzeuge kann direkt auf der Hardware des Entwicklers eine vorübergehende SINIX-Entwicklungsversion erstellt und funktionell verifiziert werden (z.B. unter Zuhilfenahme eines Analysators zur Code-Überdeckung während des Funktionsstests).

gTSIMX

Ein grafischer Terminalsimulator, der an interaktive Anwendungen angeschlossen werden kann und diese dann automatisch mit einem SOLL-/IST-Vergleich zum Ablauf bringt. gTSIMX wird sowohl zum Test von Bedienoberflächen als auch zur Beobachtung der Leistungsfähigkeit eingesetzt.

ATIX

Ein automatisches Testsystem, das etwa 20.000 Testfälle mit insgesamt mehr als drei Millionen LOCs (Lines of Source Code) verwaltet. ATIX bietet eine komfortable Bedienoberfläche, um Testfälle auszuwählen und Testergebnisse abzufragen.

PULS

Ein Fehler-Management-System zur weltweiten Verwaltung aller Problemmeldungen während des Lebenszyklus eines Produkts. PULS stellt komfortable Recherchemöglichkeiten und Trendübersichten zur Verfügung.

ProTeIn

Eine Termindatenbank, die alle für ein Projekt relevanten Termine einschließlich notwendiger externer Zulieferungen verwaltet und Übersichten zur Terminüberwachung erstellt.

Nach der Kundenfreigabe wird die Anzahl der Problemmeldungen zu einem Produkt regelmäßig kontrolliert und im Detail analysiert. Die Testwerkzeuge werden entsprechend erweitert, so daß ein kontinuierlicher Qualitätsfortschritt sichergestellt ist.

Testmethodik

Die Testmethodik bei der SINIX-Entwicklung ist gekennzeichnet durch einen mehrstufigen Prozeß, der dem jeweiligen Entwicklungsstand angepaßt wird. Alle Werkzeuge sind so konzipiert, daß sie weitgehend automatisch ablaufen. In der Regel prüfen sie das Testergebnis selbst und geben nur im Fehlerfall die für eine Diagnose notwendigen Informationen aus. Das Testsystem ATIX steuert den Start der einzelnen Werkzeuge, indem es den gewünschten Testbereich kennzeichnet und ihren Ablauf überwacht. Da ein Client/Server-Konzept verwendet wird, kann das Testsystem schnell und ohne großen Aufwand aktiviert werden.

Ein Qualitätssicherungsplan für jedes Projekt legt fest, zu welchem Zeitpunkt im Projektverlauf ein bestimmter Test erfolgen soll. Die Leistungsfähigkeit der vorhandenen Werkzeuge erfordert aus Gründen der Effizienz eine genau aufeinander abgestimmte Planung. Diese muß möglichst redundanzfrei und trotzdem vollständig sein. So beginnt die Testphase eines Produkts letztendlich mit dem Test individueller Funktionen und wächst sukzessive bis zum Test komplexer Hardware- und Software-Konfigurationen. Der Test komplexer Konfigurationen stellt insbesondere das Zusammenwirken der Produkte sicher - auch unter Grenzwertbedingungen und in einem heterogenen Netzverbund.

International standardisierte Testsuiten dienen dazu, einen objektiven Eindruck von der Qualität eines Produkts zu gewinnen. Verschiedene Testsuiten werden in unseren anerkannten Testumgebungen eingesetzt. Im SINIX-Umfeld sind die wichtigsten:

- SVVS, die UNIX System V Validation Suite
- VSX, die XPG-Testsuite zum Nachweis der X/Open-Konformität, der Voraussetzung für den Erhalt des XPG-Warenzeichens
- OSI-Testsuiten für Transportprotokolle im LAN- und WAN-Bereich
- offizielle Testsuiten für Compiler und OSF/Motif

Als Ergänzung dazu dienen weitere eigene Testfolgen dem Nachweis der Konformität zu anderen garantierten Schnittstellen wie dem SINIX API. Durch die Mitarbeit in internationalen Gremien, die sich mit den Themen Qualität und Konformität zu bestehenden Standards beschäftigen, forciert Siemens Nixdorf die Weiterentwicklung solcher Testfolgen und setzt sich für deren Standardisierung ein. Ein Beispiel hierfür ist die OSI-Testfolge, die Siemens Nixdorf im Rahmen einer Ausschreibung der Europäischen Gemeinschaft entwickelte.

Um Leistungsvergleiche zwischen unseren Systemen und denen des Mitbewerbs anstellen zu können, bringt Siemens Nixdorf eine Vielzahl anerkannter Benchmarks (TPC, SPEC, AIM, Dhrystone, usw.) auf seinen Systemen zum Ablauf und veröffentlicht die gemessenen Werte über die entsprechenden Gremien. Falls ein Kunde für eine leistungskritische Anwendung eine optimale Ausnutzung seiner Systemressourcen erzielen möchte, kann er auf die Konfigurationsempfehlungen eines Performance-Handbuchs oder ergänzend dazu auf verschiedene Werkzeuge zur Leistungsanalyse (z.B. TRACEX) zurückgreifen.

Um die Qualität und Leistungsfähigkeit einer Software-/Hardwarekonfiguration zu verifizieren, die speziell auf einen Kunden und seine spezifische dialogorientierte Anwendungssoftware zugeschnitten wurde, kann der grafische Terminalsimulator gTSIMX verwendet werden. So können Dialoge gesteuert und ein oder mehrere Anwender entweder auf real vorhandenen Terminals oder aber virtuellen Terminals simuliert werden. Eine derartige Simulation wurde von Siemens Nixdorf schon häufig als Dienstleistung beim Kunden für dessen spezielle Anwendungen durchgeführt.

Die Gesamtheit der von Siemens Nixdorf eingesetzten Test- und Kontrollwerkzeuge, ihre Weiterentwicklung sowie eine Testplanung, die auf Vollständigkeit zielt, garantieren zusammen die kontinuierlich verbesserte Qualität und eine ständig sinkende Anzahl verbleibender Fehler im Verlauf eines Projekts. Die Statistiken, die von den verschiedenen Testwerkzeugen erstellt werden, werden vom Management dazu benutzt, Jahr für Jahr noch anspruchsvollere Qualitätsziele festzusetzen.

SINIX-Dokumentation

Die hohe Qualität der Dokumentation ist eines der wesentlichen Merkmale von SINIX. Sowohl das SINIX-Betriebssystem als auch die vielen Software-Produkte, die auf SINIX aufsetzen, werden mit einer vollständigen Anwenderdokumentation ausgestattet. Die Handbücher werden im allgemeinen in deutscher und englischer Sprache produziert. In vielen Fällen werden sie jedoch auch in andere Sprachen übersetzt. Alle Dokumentationen sind einheitlich aufgebaut und weisen ein einheitliches Corporate Design auf.

Dokumentation wird bei Siemens Nixdorf von technischen Redakteuren, Experten für die Vermittlung technischer Information, in enger Zusammenarbeit mit der Software-Entwicklung erstellt. So wird gewährleistet, daß die Dokumentation folgende Anforderungen erfüllt:

- Sie ist fachlich korrekt und vollständig,
- optimal auf die jeweilige Zielgruppe ausgerichtet,
- verständlich,
- sowie termingerecht zur Lieferfreigabe eines Software-Produkts verfügbar.

Um die fachliche Korrektheit und Vollständigkeit der Dokumentation sicherzustellen, wird die Dokumentation einer sorgfältigen fachlichen Überprüfung unterzogen, an der Entwickler, Tester sowie Produkt- und Systemplanung beteiligt sind. Zusätzlich wird die Dokumentation im Lektorat von speziell dafür ausgebildeten Mitarbeitern dahingehend überprüft, ob sie verständlich und konsistent aufgebaut ist und der anvisierten Zielgruppe entspricht.

Die hohe Qualität der SINIX-Dokumentation ist schon mehrmals durch eine objektive, externe Bewertung bestätigt worden. So wurde etwa die Dokumentation für eine Reihe von SINIX-Produkten in Rezensionen in der Fachpresse äußerst positiv beurteilt. Eine Reihe von Titeln der SINIX-Dokumentation wurde von namhaften Verlagen in ihr Programm übernommen. Darüber hinaus sind mehrere Titel der SINIX-Dokumentation bei internationalen Wettbewerben von der amerikanischen „Society for Technical Communication“ ausgezeichnet worden.

Aufbau und Gliederung der SINIX-Dokumentation entspricht dem De-facto-UNIX-Standard. So kann sich jeder, der mit UNIX vertraut ist, schnell in der SINIX-Dokumentation zurechtfinden. Damit wird der Idee der offenen Systeme Rechnung getragen. Inhaltlich geht die Information jedoch weit über die des Standard-UNIX hinaus. Die Beschreibungen der SINIX-Kommandos etwa werden durch zusätzliche Beispiele erweitert, um die Verständlichkeit und Anwenderfreundlichkeit der SINIX-Dokumentation zu erhöhen.

Zur Dokumentation des SINIX-Betriebssystems gehört auch das Handbuch *Master Index*, ein übergreifendes, vollständiges Gesamt-Stichwortverzeichnis für die wesentlichen Handbücher des Dokumentationspakets. Es ermöglicht dem Benutzer, Informationen ungeachtet des großen Umfangs der SINIX-Dokumentation schnell aufzufinden.

Die Dokumentation des Betriebssystems SINIX und seiner Aufsatzprodukte besteht nicht nur aus den traditionellen Handbüchern auf Papier. Dokumentation wird auch Online am Rechner zur Verfügung gestellt. Siemens Nixdorf bietet zudem neue Technologien wie Hypertext oder kontextsensitive Hilfe.

Man Pages

SINIX unterstützt das Standard-UNIX Kommando *man*, das dem Anwender erlaubt, Online auf Informationen der Referenzhandbücher zugreifen zu können. In SINIX wird der Text dieser sogenannten *Man Pages* aus der gleichen Quelldatei wie die Referenzhandbücher erzeugt. Papierdokumentation und Online-Dokumentation sind damit einheitlich und vollständig. Siemens Nixdorf bietet auch zusätzliche *Man Pages* im Standardformat an, um Aufsatzprodukte zu unterstützen.

Online-Hilfe

Neben den Man Pages stellt SINIX auch Online-Hilfen zur Verfügung. Dies geschieht in einigen Fällen über Fremdprodukte wie etwa OSF/Motif. In anderen Fällen wird die Online-Hilfe direkt von Siemens Nixdorf als Bestandteil eines eigenen Software-Produkts angeboten. Das Produkt SINIX/windows erweitert die Online-Hilfe um kontextsensitive Hilfe und hypertext-ähnliche Verknüpfungen. Kontextsensitive Hilfe ermöglicht dem Anwender, Informationen über ein bestimmtes Objekt auf dem Bildschirm (z.B. eine Menüoption,) zu erhalten. Die hypertext-ähnlichen Verbindungen erlauben dem Anwender, sich durch eine Reihe von Hilfetexten zu bewegen, indem er hervorgehobene Wörter auswählt.

Ausblick

Siemens Nixdorf wird seine Anstrengungen fortsetzen, um sicherzustellen, daß die Dokumentation den effektiven Einsatz von SINIX-Systemen und SINIX-Software-Produkten erleichtert. Weiterhin werden hohe Qualitätsanforderungen an Papierdokumentation, Man Pages und Online-Hilfen gestellt und zusätzliche Techniken getestet werden, um dem Anwender Dokumentation zur Verfügung zu stellen. Zur Zeit ist die Präsentation der SINIX-Dokumentation auf CD-ROM mit einem Retrievalsystem und grafischer Bedienoberfläche in Vorbereitung.

10 Zukünftige Entwicklungen

Überblick

Zukünftige Entwicklungen lassen sich nicht mit Sicherheit vorhersehen. Einige Dinge können wir jedoch schon sehen, denn wir wissen natürlich, um wie viele Jahre die Forschung in den Bereichen Informatik und Computertechnik mittlerweile den bereits auf dem Markt befindlichen Systemen voraus ist. Die Forschungsarbeit in den Laboren gibt uns einen guten Einblick in die Möglichkeiten, die uns die Datenverarbeitung im kommerziellen Einsatz in Zukunft geben wird. Es ist naturgemäß nur schwer vorherzusagen, welche dieser Technologien sich tatsächlich in die Praxis umsetzen lassen, und aufgrund welcher ökonomischen Gegebenheiten sie sich auf dem Markt durchsetzen werden.

Bei allen Planungen muß man also die Entwicklungsmöglichkeiten der Zukunft miteinkalkulieren. Drei technologische Bereiche werden unserer Meinung nach in der kommerziellen Datenverarbeitung eine entscheidende Rolle spielen: kooperative Datenverarbeitung, objektorientierte Verarbeitung und Multimedia-Fähigkeit. In diesem Kapitel erhalten Sie einen Überblick über diese drei Bereiche.

Kooperative Datenverarbeitung

Der hier verwendete Begriff der kooperativen Datenverarbeitung bezieht sich auf drei Ebenen: die Bediener- oder Anwendungsebene, die Betriebssystemebene und die Hardwareebene. Auf Bedienerebene wird die weite Verbreitung von verteilten Anwendungen dazu führen, daß unterschiedliche Prozesse (Clients und Server) auf verschiedenen Rechnern miteinander kooperieren. Auf der Betriebssystemebene wird es die sogenannte Mikrokerneltechnologie ermöglichen, daß die unterschiedlichen Betriebssysteme in eine Reihe von kooperierenden Prozessen aufgeteilt werden können. Auf Hardwareebene werden mehrere Prozessoren in einem Rechner in ganz unterschiedlichen Konfigurationen kooperativ miteinander arbeiten können, indem Prozesse auf Betriebssystem- und Anwendungsebene parallel bedient werden.

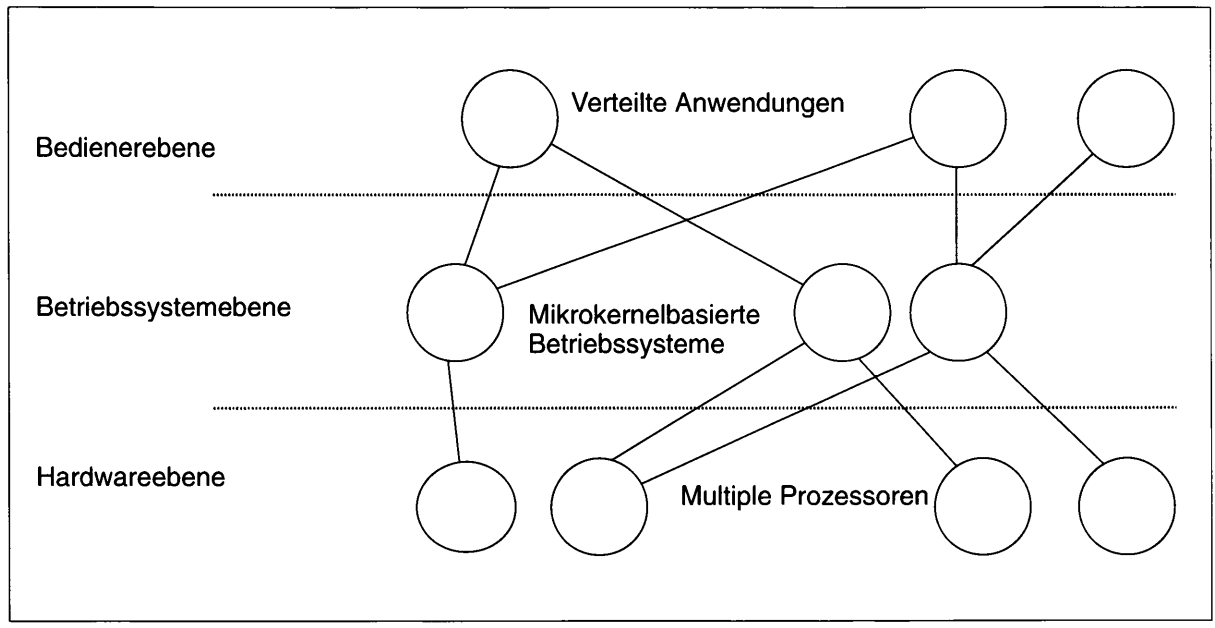


Abbildung 10-1: Die kooperative Datenverarbeitung umfaßt drei verschiedene technologische Gebiete auf drei verschiedenen Ebenen.

Kooperative Datenverarbeitung bietet deutliche Vorteile, die im folgenden kurz exemplarisch dargestellt werden:

- Die kooperative Datenverarbeitung erlaubt die gemeinsame Nutzung von Ressourcen und trägt damit zu einer erheblichen Kostensenkung bei. Wenn z.B. die Leerlaufzeiten eines Prozessors vermindert werden können, kann man ohne Mehraufwand und Kosten ansonsten vergeudete Zeit einsparen.
- Durch die kooperative Datenverarbeitung kann die Leistungsfähigkeit von Computern kostengünstig durch einfaches Hinzufügen von vorgefertigten und damit billigen Bauteilen in Modularbauweise erhöht werden. Man muß kein weiteres großes und teures System anschaffen, sondern kann schrittweise kostengünstige Bauteile in das System einfügen: z.B. kann man eine weitere Prozessorbaugruppe in einen vorhandenen Rechner einbauen oder einen zusätzlichen kleineren Rechner in ein Netzwerk einhängen.
- Die kooperative Datenverarbeitung erleichtert die Teamarbeit und die Arbeit an gemeinsamen Projekten. Arbeitsgruppen können gemeinsame Ressourcen einsetzen, z.B. auf gemeinsam genutzte Dateien zugreifen, Echtzeitkommunikation betreiben, sich Computerarbeitsplätze teilen usw.
- Die kooperative Datenverarbeitung steigert mit geringen Kosten die Verfügbarkeit, indem Dienstleistungen mehrfach angeboten werden. Wenn ein Anbieter von Dienstleistungen ausfällt, kann ein anderer dafür schnell einspringen, ohne daß dadurch der Betrieb gestört oder ein Anwender etwas bemerken würde.

Technologien auf Anwendungsebene: Verteilte Verarbeitung

Siemens Nixdorf geht davon aus, daß die kooperative Datenverarbeitung auf Anwenderebene vor allem auf der Grundlage von DCE entwickelt werden wird, da DCE von den meisten Computerherstellern und Softwarehäusern an ihre jeweiligen Systeme angepaßt wurde. DCE ist damit zum De-facto-Industriestandard geworden und X/Open ist dabei, DCE zum Common-Applications-Environment-Standard (CAE) zu erheben.

Vorteile der kooperativen Datenverarbeitung für den Anwender

Die wichtigsten Vorteile der kooperativen Datenverarbeitung auf Anwenderebene mit DCE können folgendemaßen skizziert werden:

- nicht sichtbare örtliche Verteilung
- Verfügbarkeit
- Anpassungsfähigkeit
- Integrierte Sicherheit
- Einfache Bedienung

Nicht sichtbare örtliche Verteilung

Den Benutzern von DCE bleibt der Standort eines Rechners, der bestimmte Dienstleistungen bereitstellt, verborgen. Durch die besonderen Bindemechanismen von DCE und dem Name-Service bleiben die DCE-Dienste in hohem Maße ortsunabhängig.

Verfügbarkeit

Verteilte Systeme müssen unabhängig von Netzunterbrechungen oder Fehlern auf fernen Rechnern arbeiten können. DCE unterstützt zur Erhöhung der Verfügbarkeit die Mehrfachhaltung des Cell-Directory-Servers und des Security-Servers. Durch eine Mehrfachhaltung von Anwendungsservern kann eine zusätzliche Steigerung in der Verfügbarkeit erreicht werden.

Anpassungsfähigkeit

Netzwerke der Zukunft werden um ein Vielfaches größer und geographisch weitläufiger sein als heute. Folgendes Szenario ist keine Utopie: Fast alle PCs verfügen über eine Netzanbindung und Metropolitan Area Networks (MAN) oder Wide Area Networks (WAN) sind weit verbreitet. Hunderte von parallel gehaltenen Servern können Tausenden von Clients ihre Dienste anbieten. Einige Server müssen innerhalb kürzester Zeit vielen Hundert Clients ihre Dienste zur Verfügung stellen, und einige Clients müssen in ebenso kurzer Zeit mit Hunderten von Servern Kontakt aufnehmen. Die einzelnen Komponenten von DCE sind so angelegt, daß sie sich ohne Schwierigkeiten großen Netzwerken anpassen können, ohne daß dabei ein Verlust an Leistung, Zuverlässigkeit oder Flexibilität zu befürchten wäre. Da es kein Kunde schätzt, wenn sich herausstellt, daß ein Computersystem gleichsam über Nacht zu klein geworden ist und man nun von vorne anfangen muß, ist es wichtig, daß man die Möglichkeit hat, immer neue Maschinen auch von unterschiedlichen Herstellern dazuzukaufen, diese in vorhandene Netze einzubinden, Netzwerksysteme zu bilden, immer umfangreichere Anwendungen und Datenbanken einzusetzen, ohne dabei etwas wegwerfen oder neu aufbauen zu müssen. DCE garantiert diese Erweiterungsfähigkeit.

Sicherheit

Eine der wichtigsten Voraussetzungen für verteilte Verarbeitung ist die Sicherheit. In DCE sind die neusten Forschungsergebnisse (Identifizierung und Autorisierung) eingeflossen, um ein Höchstmaß an Sicherheit zu gewährleisten.

Einfache Bedienung

DCE ist einfach zu bedienen, da sich der Anwender fast um nichts kümmern muß. Durch seinen Name-Service erreicht es DCE, daß es keine Rolle spielt, auf welchem Netzrechner eine bestimmte Ressource verfügbar ist, denn für den Anwender spielt es keine Rolle, auf welchem Rechner die Daten liegen und welcher Rechner diese gerade bearbeitet. Kein Anwender möchte sich mit dem komplexen Aufbau des zugrundeliegenden physikalischen Netzes befassen müssen, auch die komplizierten Protokolle und die ungeheure Rechenleistung interessieren kaum. Anwender möchten ohne Aufwand an ihre Daten kommen, sie wollen dafür keine komplizierten und schwer zu handhabenden Prozeduren verwenden müssen.

Die Mehrfachhaltung der Ressourcen in DCE ist für den Anwender nicht sichtbar. Eine beliebige Anzahl von Anwendern kann dieselbe Ressource anfordern und keiner der Anwender wird bemerken, daß er nicht der einzige ist, der dieselbe Dienstleistung in Anspruch nimmt.

In einem korrekt aufgebauten System, das auf DCE beruht, werden auftretende Fehler durch Mehrfachhaltung der Ressourcen keinen Schaden anrichten und nicht in Erscheinung treten. Ein auftretendes Problem eines Prozesses, eines Rechners oder der Netzwerkverbindung kann also einen Anwender nicht in der Fortführung seiner Arbeit behindern.

Siemens Nixdorf fördert die verteilte Verarbeitung

Siemens Nixdorf spielt eine wesentliche Rolle bei der Entwicklung und Vermarktung der verteilten Verarbeitung in Form des Softwareproduktes **D**istributed **C**omputing **E**nvironment (DCE) der **O**pen **S**oftware **F**oundation (OSF). Siemens Nixdorf ist einer der Mitbegründer und Sponsoren der Open Software Foundation. Das Produkt DIR-X von Siemens Nixdorf wurde von OSF als Grundlage des DCE-basierten Global Directory Service ausgewählt. OSF hat Siemens Nixdorf als Lieferanten der Referenzimplementierung von DCE auf UNIX SVR4 bestimmt. Siemens Nixdorf und die **U**NIX **S**ystem **L**aboratories (USL) sind in einem Kooperationsvertrag übereingekommen, daß USL die DCE-Referenzimplementierung verwendet und im expandierenden UNIX-SVR4-Markt unterstützt. Siemens Nixdorf wird auch weiterhin DCE aktiv fördern und daran arbeiten, die Anzahl der mit DCE verfügbaren Anwendungen zu erhöhen und damit zur weiteren schnellen Verbreitung der verteilten Verarbeitung auf Anwenderebene erheblich beizutragen.

Neue Technologien der Betriebssysteme

Die Betriebssysteme der neuen Generation in Mikrokerneltechnologie teilen die logische Funktionalität des Kernels eines traditionellen Betriebssystems in funktionale Objekte. Diese funktionalen Objekte bilden unabhängige, miteinander kommunizierende Einheiten. Ein gegenüber herkömmlichen Systemen vergleichbar kleiner Teil des Betriebssystemkerns, der sogenannte Mikrokernel, enthält eine Sammlung von grundlegenden Mechanismen, die den Betrieb eines Betriebssystems ermöglichen. Weitergehende Funktionen, die für besondere Arbeitsumgebungen in einem Betriebssystem benötigt werden, können dann außerhalb des Mikrokernels realisiert werden; dadurch bleiben sie vom Mikrokernel unabhängig. Der Mikrokernel bildet die einzig mögliche Schnittstelle zur Hardware. Seine Strukturierung in hardwareabhängige und hardwareunabhängige Teile macht seine hohe Modularisierung und Erweiterungsfähigkeit und damit einfache Portierbarkeit deutlich.

Werden weitere Funktionselemente außerhalb des Mikrokernels benötigt, um eine komplexe Betriebssystemumgebung aufzubauen (z.B. eine SINIX-Umgebung), so werden diese in Form von Servern bereitgestellt. Die Server repräsentieren dann funktionale Objekte, die ihre Dienste in Form von Client-Server-Verbindungen anbieten. Auf diese Weise wird der volle Funktionsumfang eines Betriebssystems erreicht, indem einzelne Elemente zusammengefügt werden. Die einzelnen Komponenten sind die Module, aus denen dann ein ganzes System aufgebaut wird. Man spricht in diesem Zusammenhang von einem skalierbaren System.

Betriebssysteme in Mikrokerneltechnologie unterscheiden sich von traditionellen Betriebssystemen vor allen Dingen in der Speicherverwaltung. Die einzelnen Speicherregionen werden in sogenannte Speicherobjekte aufgeteilt. Die Verwaltung dieser Speicherobjekte im Hauptspeicher wird vom Mikrokernel übernommen. Ein spezieller Speichermanager (Speicher-Server) ist für die nicht speicherresidenten Abbildungen (z.B. auf der Festplatte) zuständig. Der Mikrokernel arbeitet unabhängig von den sekundären Abbildungen in den Speicherregionen, für die ausschließlich der Speichermanager zuständig ist. Auf diese Weise können z.B. verschiedene Paging-Algorithmen, Shared-Memory über Rechnergrenzen hinweg und Mechanismen zur besseren Fehlertolerierung in den Mikrokernel und in Anwenderprogramme eingebaut werden, ohne daß dies nach außen sichtbar wird.

Aus den oben genannten Gründen lassen sich die Vorteile der Mikrokerneltechnologie wie folgt zusammenfassen:

- Wegen der einfachen Portierbarkeit der Mikrokernel auf andere Prozessor- oder Speicherarchitekturen wird auch der Umstieg auf neue Hardwarearchitekturen erleichtert, d.h. besseres Time-to-Market.
- Die eingebaute Fähigkeit zur Parallelverarbeitung ermöglicht eine bessere Ausnutzung von Multiprozessorrechnern, als dies bisher möglich war. Dies ermöglicht auch den Einsatz von massiv parallelen Architekturen.
- Die interne Strukturierung (also Unterteilung in Funktionsgruppen) in unabhängige, autonom arbeitende Einheiten (Server) bietet die Möglichkeit, ein System rasch durch neue Funktionen zu erweitern und damit einfach und flexibel zukünftige Anforderungen erfüllen zu können.
- Es ist möglich, auf einem Rechner mehrere Versionen des gleichen Betriebssystems ablaufen zu lassen, indem einfach die jeweiligen Server geladen werden. Der Kunde kann so eventuell auftretenden Kompatibilitätsproblemen mit älteren Anwendungen beim Übergang zu einer neuen Betriebssystemversion aus dem Weg gehen und den Aufwand für eine Anpassung der Anwendung sparen.
- Auf einem Rechner können die verschiedensten Betriebssysteme nebeneinander betrieben werden, z.B. SINIX, MS-DOS, MS-Windows, OS/2 usw. Die Kunden erhalten dadurch eine größere Freiheit bei der Auswahl der Anwendungen, in die sie investieren.
- Indem einzelne Prozesse in Clients und Server aufgeteilt werden können, erhöht sich u.a. die Zuverlässigkeit und Stabilität des gesamten Systems beträchtlich. Die Ausfallzeit eines Systems kann erheblich verringert werden, da Server mehrfach vorhanden sein können, ohne daß dies vom Client bemerkt wird.

Die derzeit verbreitetsten Mikrokernelsysteme sind OSF/1 MK von OSF (basierend auf dem Betriebssystem Mach der Carnegie Mellon University), das System CHORUS von CHORUS Systèmes und Windows NT von Microsoft.

Einem Server (Management eines SINIX-Prozesses, des Dateisystems usw.) kann man Betriebssystemaufgaben (z.B. SINIX) übertragen. Außerdem geht man heute dazu über, Server in sogenannte generische und spezifische Server zu unterteilen. Innerhalb eines generischen Servers werden Aufgaben von allgemeinem Interesse ausgeführt, z.B. Name-Server, Identifikations-Server oder Datei-Server. Innerhalb eines spezifischen Servers werden Aufgaben erledigt, die zu einer Betriebssystemumgebung (z.B. SINIX-, Windows-, DOS-, und OS2-Server) gehören. Damit wird ein hoher Grad an Flexibilität erreicht. Skalierbare Systeme können auf diese Weise aus einem Mikrokernel und aus verschiedenen, vom Server bereitgehaltenen Funktionen erstellt werden. Durch Verwendung geeigneter Mechanismen ist es deshalb möglich, Programme, die für ein bestimmtes Betriebssystem (z.B. SINIX) entwikkelt wurden, durch einen Mikrokernel auf einem entsprechenden Betriebssystemserver binärkompatibel ausführen zu lassen. Die einzige Voraussetzung dafür ist, daß das Programm für den verwendeten Prozessortyp compiliert wurde.

Mikrokerneltechnologie bei Siemens Nixdorf

Siemens Nixdorf ist sehr daran interessiert, die Mikrokerneltechnologie zu fördern. Dies bedeutet nicht nur die Förderung der theoretischen Forschungsarbeit, sondern auch die praktische und experimentelle Arbeit. Eine Forschungsgruppe von Siemens Nixdorf hat z.B. die Funktionalität von UNIX SVR4 (SINIX) in Form eines Servers auf das OSF/1 MK-System übertragen. Gleichzeitig wurde die technische Durchführbarkeit der Portierung dieses Servers auf Windows NT untersucht. Gegenwärtig wird der Server von Siemens Nixdorf als Teil eines Forschungsprojekts der Europäischen Gemeinschaft auf das System CHORUS portiert. Diese Arbeit dient dazu, den technologischen Vorsprung von Siemens Nixdorf auf dem Gebiet der Entwicklung, des Tests, der Pflege und des Einsatzes solcher Systeme zu demonstrieren. Die SINIX-Entwicklung fördert diese weltumspannenden Aktivitäten und trägt stets dafür Sorge, das nötige Wissenspotential zur Verfügung zu haben. Ziel ist es, diese neuen Technologien in die Weiterentwicklung von SINIX einfließen zu lassen.

Ein Betriebssystem bildet die Brücke zwischen den Anwendungen und der zugrundeliegenden Hardware. Deshalb muß ein Betriebssystem auch schnell an die sich verändernden Computerarchitekturen anpaßbar sein. Diese Anpassungsfähigkeit führt zu einer Kostensenkung für Entwicklung, Wartung und Service und ermöglicht eine schnellere Verfügbarkeit neuer Technologien, z.B. auf dem Gebiet der Prozessortechnik. Für den Kunden ist es wünschenswert, daß die verschiedenen Rechnerarchitekturen, seien es Einzelprozessorrechner, Multiprozessorrechner, massiv parallele Systeme oder über Netzwerke gekoppelte Betriebssysteme bedient werden können. Aus diesem Grund gibt es die Forderung nach sogenannten skalierbaren, den Kundenanforderungen anpaßbare Systeme. Man erreicht damit Systemstrukturen, aus deren Bestandteilen ein komplettes System gebildet werden kann, das dann unter gegebenen Voraussetzungen erstellt, installiert und angepaßt wird. Damit ist gewährleistet, daß der Kunde immer über ein an die technischen Neuerungen angepaßtes System verfügt. Vorhandene langfristige Investitionen bleiben damit über einen langen Zeitraum gesichert, vorhandene Anwendungen können schneller und flexibler an neue Gegebenheiten angepaßt werden.

Siemens Nixdorf ist stets darum bemüht, seinen Kunden immer den neusten Stand der Technik zu liefern, sobald dies wirtschaftlich vertretbar ist. Unsere Kunden haben in Anwendungen investiert, die auf Aussagen im SINIX API beruhen. Siemens Nixdorf wird das SINIX API auch für die verschiedenen Mikrokernelsysteme unterstützen. Das erlaubt unseren Kunden, neue Computerarchitekturen zu nutzen, ohne dabei schon getätigte Investitionen in SINIX-Systeme zu verlieren. Da mikrokernelbasierte SINIX-Systeme das SINIX API anbieten werden, brauchen schon existierende Anwendungen nicht umgeschrieben zu werden, damit diese mit der neuen Architektur und mit Anwendungen, die speziell für eine Mikrokernelarchitektur neu entwickelt wurden, zusammen ablauffähig sind.

Modularisierte Komponenten eines Mikrokernels können zu einem späteren Zeitpunkt ohne weiteres durch neuere ersetzt werden, damit neuere Rechnerarchitekturen und Server unterstützt werden. Weitere Module können hinzugefügt werden, um zusätzlich zum SINIX-Grundsystem andere Systemumgebungen anbieten zu können, beispielsweise die Unterstützung von Windows zusätzlich zu SINIX oder von herstellerspezifischen Systemen. Es folgt ein kurzer Überblick über die Forschungstätigkeit von Siemens Nixdorf auf dem Gebiet der Mikrokerneltechnologie.

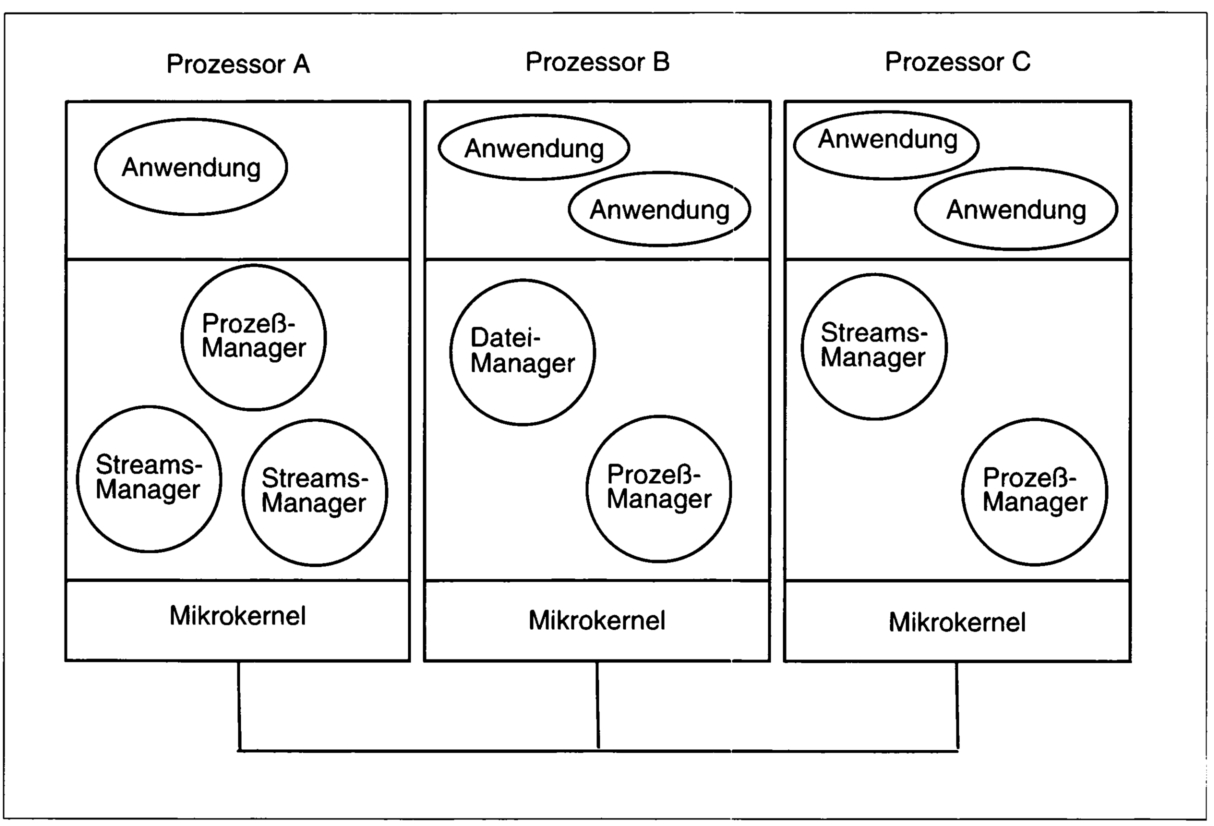


Abbildung 10-2: Die Funktionen eines herkömmlichen Betriebssystems, die nicht in den Mikrokernel integriert sind, können auf verschiedene Prozessoren verteilt werden. Das Beispiel zeigt die Konfiguration eines CHORUS-Systems mit drei Prozessoren, die über Netz oder Bus miteinander verbunden sein können.

Mach 3.0

Siemens Nixdorf hat einen SINIX-Server entwickelt, der Mach 3.0 unterstützt. Dieser Server ist modularer Systembestandteil von Mach 3.0 und arbeitet im Usermodus des Prozessors als Mach-Anwendung. Dieses Subsystem ist 100% binärkompatibel für existierende SINIX-Anwendungen. Das zeigt deutlich, wie die Mikrokernelarchitektur die Erwartungen in Bezug auf Kompatibilität erfüllt. Siemens Nixdorf hat hiermit ein SINIX-Subsystem für Mach 3.0 verfügbar, das im Interesse der Siemens-Nixdorf-Kunden neue Wege für die Weiterentwicklung der Betriebssysteme in der Mach-3.0-Welt eröffnet. Die aus dieser Entwicklung gewonnene Erfahrung bildet eine hervorragende Basis für die Weiterentwicklung von Mikrokernelsystemen.

Das OSF Forschungsinstitut leistet hier weiterführende Arbeit. Ausgehend von einem Mach-3.0-Mikrokernel wurde OSF/1-AD entwickelt. Diese Arbeit wurde von Siemens Nixdorf begutachtet. Sobald diese Technologie stabil und für den kommerziellen Einsatz verfügbar ist, können wir eine SINIX-basierte Implementation entwickeln.

Chorus

Zusammen mit den UNIX System Laboratories und anderen europäischen Firmen aus dem Bereich Datenverarbeitung nimmt Siemens Nixdorf am Projekt Ouverture teil. Dieses anspruchsvolle Entwicklungsprojekt ist dazu angelegt, die ES/MP-Version des UNIX-Betriebssystems von USL auf den Chorus-Mikrokernel zu verteilen. Sobald dieses Projekt erfolgreich abgeschlossen ist, wird es in Zukunft möglich sein, SINIX-Anwendungen auf verteilten und fehlertoleranten Systemen ablaufen zu lassen.

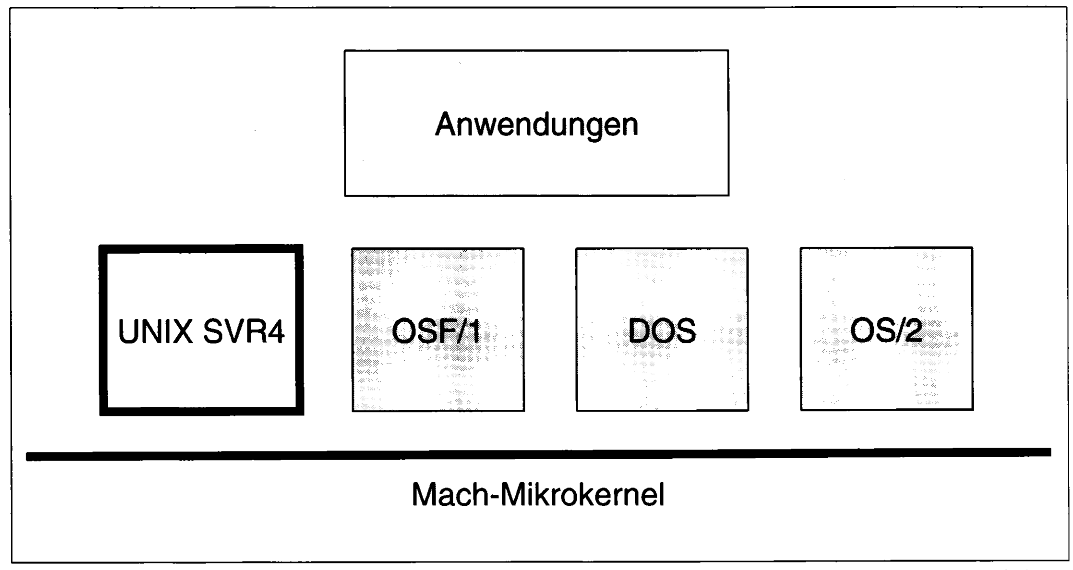


Abbildung 10-3: Man kann verschiedene Betriebssystem-Server (man spricht auch von „Personalities") auf einen Mikrokernel aufsetzen. Im Rahmen eines Forschungsprojekts wurde UNIX SVR4 in einen Mach-Mikrokernel eingebunden.

Neue Technologien auf dem Hardwaresektor

Als treibende Kraft hinter den Systementwicklungen steht der Wunsch nach immer größerer Leistung. In den letzten Jahren konnten erstaunliche Entwicklungssprünge bei der Entwicklung neuer Prozessortechnologien verzeichnet werden. Zudem werden Leistungssteigerungen nicht allein dadurch erzielt, daß bessere Prozessoren entwickelt werden, sondern auch durch die Kombination von mehreren Prozessoren in einem Rechner. Theoretisch könnte man, wenn man nur eine genügend große Zahl von Prozessoren verwendete, die Rechenleistung schier ins Unermeßliche steigern. Leider sieht die Wirklichkeit anders aus, denn man erreicht ab einer gewissen Größe Leistungsgrenzen, die durch die Rechnerarchitektur und das verwendete Betriebssystem vorgegeben sind. Beläßt man die Zahl der Prozessoren aber in einem durch das System vorgegeben Rahmen, kann man mit einem Multiprozessorsystem durch einfaches Hinzufügen von neuen Prozessoren erhebliche Leistungssteigerungen erreichen. Trotz steigender Ansprüche braucht man nicht auf ein anderes Computersystem umzusteigen.

Computersysteme mit mehreren parallel betriebenen Prozessoren kann man in zwei Gruppen unterteilen: SIMD- (**s**ingle **i**nstruction **m**ultiple **d**ata) und MIMD-Systeme (**m**ultiple **i**nstruction **m**ultiple **d**ata). SIMD beschreibt eine Architektur, in der alle vorhandenen Prozessoren gleichzeitig die gleichen Operationen für unterschiedliche Daten ausführen. Dieses Verfahren wird vor allem im wissenschaftlichen Bereich angewandt, wo komplizierte Gleichungen mit großen Datenmengen gelöst werden müssen. MIMD beschreibt eine Architektur, in der die einzelnen Prozessoren unterschiedliche Programme ausführen. Dieses Verfahren ist im allgemeinen besser im kommerziellen Bereich einsetzbar. Bei der MIMD-Architektur kann man zwei Varianten unterscheiden: Es gibt Systeme, die Distributed Memory (verteilte Speicher) und solche, die Common Memory (gemeinsam nutzbare Speicher) verwenden. Distributed Memory bedeutet, daß den einzelnen Prozessoren eigene Speicherbereiche zugeordnet sind und Daten untereinander über extrem schnelle Verbindungen (sog. Interconnects) austauschen, während Common Memory bedeutet, daß alle vorhandenen Prozessoren auf einen einzigen gemeinsamen Speicherbereich zugreifen. Die meisten bisherigen UNIX-Systeme auf Multiprozessorbasis sind solche mit Common-Memory-Architektur.

Sobald in einem Multiprozessorsystem viele Prozessoren gleichzeitig über einen gemeinsamen Bus auf Daten zugreifen wollen, wird der Bus schnell zu einem Engpaß. Deshalb wird in Multiprozessorsystemen mit Hochleistungs-Cache-Speichern gearbeitet, damit die Häufigkeit der Zugriffe auf den Hauptspeicher drastisch reduziert werden können. So kann man die Anzahl der Datenzugriffe einer zentralen Recheneinheit auf den Speicher normalerweise auf wenige Prozent der ursprünglichen Menge verringern. Für die Datenkonsistenz muß dann die Hardware Sorge tragen.

Der große Vorteil des Common Memory liegt vor allem darin, daß man nicht erst eigene Programme für diese Art der Speicherverwaltung schreiben muß. Jedes Anwendungsprogramm, das für ein Monoprozessorsystem entwickelt wurde, ist grundsätzlich auch auf einem Multiprozessorsystem ablauffähig. Ein Programm, das für ein Monoprozessorsystem entwickelt wurde, ist auf einem System mit Distributed Memory ohne Probleme ablauffähig. Unterschiede ergeben sich bei einem Programm, das mehrere Verarbeitungseinheiten nutzt (Multithreading), die in einem gemeinsamen Speicher Daten halten und sich über Semaphore synchronisieren. Solche Anwendungen müssen an Stelle von einfachen Speicherzugriffen Meldungen an die anderen Verarbeitungseinheiten schicken.

Kleine Multiprozessorsysteme

Wenn man von kleinen Multiprozessorsystemen spricht, meint man Systeme in einer Größenordnung von zwei bis 20 oder 30 Prozessoren. Die Produktpalette von Siemens Nixdorf umfaßt derzeit zwei kleine Multiprozessorsysteme, die CISC-Linie mit Intel-Prozessor und die MIPS-Linie mit RISC-Prozessor. In den folgenden Abschnitten werden die kleinen Multiprozessorsysteme näher dargestellt.

Symmetrische Multiprozessorsysteme

Ein erfahrenes Entwicklungsteam hat das Betriebssystem SINIX so erweitert, daß es die volle Leistungsfähigkeit von Multiprozessorsystemen nutzen kann. Es handelt sich dabei um symmetrische Multiprozessorsysteme. Jeder Prozessor kann einen beliebigen Prozeß ausführen, es gibt keinen Prozessor für besondere Aufgaben. Dies gilt sowohl für die Software als auch für die Hardware. Ein gutes Beispiel für dieses Verfahren ist die Verteilung der Interrupts. Interrupts sind Unterbrechungen durch Hardwaresignale, die z.B. den Abschluß eines Datentransfers mitteilen.

In einem symmetrischen Multiprozessorsystem kann jeder Prozessor beliebige Interrupts bearbeiten. Auch in einem SINIX-Multiprozessorsystem kann jeder Prozessor die Interrupts der Hardwareuhr bedienen und so die Zeitscheiben für alle Prozesse auf dem neusten Stand halten. Die gleichmäßige Verteilung der Interrupts wird nach der Priorität der einzelnen Prozesse eines Prozessors vorgenommen. Wird einem Prozeß vom Betriebssystem eine höhere Priorität zugewiesen, wird der betroffene Prozessor dadurch seltener durch Hardwareinterrupts unterbrochen. Der Vorteil dieser Verteilung liegt in einer, durch Interrupts verursachten gleichmäßigen Auslastung der Prozessoren. Wäre nur ein Prozessor für alle Interrupts zuständig, könnte es eher zu Wartesituationen bei der Interruptbehandlung kommen. Die Prozesse in den anderen Prozessoren, die auf Interrupts warten, werden dardurch verzögert. Dadurch würde auch die gesamte Leistung des Rechners erheblich absinken, obwohl bei anderen Prozessoren noch erhebliche Ressourcen verfügbar wären.

Solche Systeme arbeiten sogar in der Boot-Phase voll symmetrisch, denn das System kann mit einem beliebigen Prozessor hochgefahren werden. Im Fehlerfall braucht keine der Prozessorbaugruppen ausgetauscht zu werden, da der Rechner einfach auf den nächsten verfügbaren Prozessor umschaltet und das Betriebssystem dann hochfährt. Diese Aufgabe wird von einer besonderen Baugruppe für Diagnosezwecke (SCED-Board) erledigt, die im Bedarfsfall Befehle an Prozessorbaugruppen senden kann. Diese Befehle können Prozessoren auch dazu einsetzen, andere

Prozessorbaugruppen ab- oder zuzuschalten. Man verwendet dazu den Kernel-Debugger. Im Fall eines Defekts oder eines Testlaufs kann ein betroffener Prozessor alle anderen Prozessoren anhalten und sich mit Hilfe des Debuggers auf der Konsole melden. Der Kernel-Debugger zeigt dann den aktuellen Zustand des betroffenen Prozessors an, damit man schnell die Probleme isolieren kann, die zu der sogenannten Race-Condition (Lauffehler; eine Situation, in der Prozessoren gegenseitig Daten zerstören) geführt haben.

Um SINIX an ein Multiprozessorsystem anzupassen, wurde der Kernel des Betriebssystems für den Parallelbetrieb ausgelegt, damit mehrere Prozessoren gleichzeitig Kernelroutinen ausführen können. Obwohl ein einzelner Anwenderprozeß zu einer Zeit nur auf einem Prozessor ablaufen kann, können andere Anwenderprozesse gleichzeitig auf andere Prozessoren ablaufen. Die einzelnen Prozesse stören sich gegenseitig nicht, da ihre Anwenderdaten voneinander unabhängig sind und sich immer in unterschiedlichen Speicherregionen befinden. Trotzdem verwenden alle Prozesse gemeinsam Kernelroutinen, die ihrerseits gemeinsam Daten nutzen. Aus diesem Grund muß der Kernel auf allen Prozessoren gleichzeitig ablauffähig (d.h. multiprozessorfähig) sein.

Wenn zwei Prozessoren versuchen, zur gleichen Zeit auf einen bestimmten Speicherplatz zu schreiben, müssen durch einen speziellen Sperrmechanismus (engl. Lock) die Zugriffe synchronisiert werden, um Inkonsistenzen zu vermeiden. Bevor ein Prozessor Systemdaten verändern darf, z.B. Vergabe einer neuen Prozeßnummer, muß dieser Sperrmechanismus in Gang gesetzt werden. Ist eine Sperre gesetzt, muß der Prozessor warten. In einem Multiprozessorsystem kann der Prozessor während dieser Wartezeit nichts anderes tun, als die gewünschte Sperre anzufordern und ständig abzufragen und zu warten, bis sie frei geworden ist. Erst dann ist der Zugriff auf den Speicherplatz erlaubt.

Bei der Entwicklung von parallelen SINIX-Systemen wurden die soeben beschriebenen Probleme außergewöhnlich gut gelöst. Die Zentralbereiche des Kernels, wie z.B. Verwaltung des virtuellen Speichers oder Dateisystemroutinen, zeigen praktisch keine durch Kollisionen bedingte Stillstandsituationen. Dies wird durch die Tatsache belegt, daß die Leistung eines Zwei-Prozessorsystems bei den meisten Anwendungen 1,8-mal über der eines Monoprozessorsystems liegt. Das ist ein eindrucksvolles Ergebnis.

Massiv Parallele Systeme

Symmetrische Multiprozessorarchitekturen sind auf Systeme mit ca. 20 bis 30 Prozessoren beschränkt. Danach bringt eine höhere Anzahl von Prozessoren keine signifikante Leistungssteigerung mehr, es sind sogar Leistungsminderungen möglich (Überlastung des Datenbus zum Hauptspeicher, Simultantreffer unter den Caches, Verfügbarkeit, usw.). Aus diesem Grund wird an neuen Lösungen gearbeitet. Massiv parallele Systeme sind große Systeme, die aus vielen preisgünstigen Mikroprozessoren bestehen. Solche Systeme binden viele Mikroprozessoren durch neuartige Kommunikationsmechanismen aneinander, um die Arbeitsbelastung des Systems unter den verfügbaren Prozessoren aufzuteilen. Die Anzahl der Prozessoren kann theoretisch in die Tausende gehen.

Um die Vorteile dieser neuen Computersysteme voll und ganz nutzen zu können, wird Betriebssystem-Software nötig, die in der Lage ist, die Systemressourcen zu verwalten und zu koordinieren. Bedeutende Änderungen des Betriebssystems sind erforderlich, um die zugrundeliegende Rechnerarchitektur zu unterstützen. Mikrokernelbasierte Betriebssysteme stellen einen weitaus flexibleren Ausgangspunkt für die Entwicklung eines modularen Betriebssystems dar, das entsprechend der Anforderungen dieser Hardware-Plattformen konfiguriert werden kann.

Massiv parallele Rechner sind derzeit ein wichtiges Thema in Forschung und Entwicklung. Sie werden von einigen Herstellern auch schon auf dem kommerziellen Markt angeboten. Diese Rechner bieten den Vorteil flexibler Abstufung aller Hardware-Elemente, so daß eine auf die unterschiedlichsten Anwendungsfälle zugeschnittene Konfiguration zur Verfügung gestellt werden kann, die durch Hinzufügen zusätzlicher Prozessoren noch erweiterbar ist.

Es gibt massiv parallele Computer mit einfachem Netzwerk, z.B. LANs. Für Anwendungen ist die Verteilung darin direkt sichtbar. Ohne großen Anpassungsaufwand können bereits bestehende Anwendungen aus diesem Typ so gut wie keine Vorteile ziehen. Vorteilhaft ist jedoch ein anderer Typ massiv paralleler Systeme, der die Adressierung von verteilten Speichern im Cluster transparent mit Hilfe von Verzeichnissen verbirgt (Distributed Shared Memory). Diese Architektur eignet sich bestens für Vielzweck-Systeme wie SINIX und seine Anwendungen, da diese Anwendungen ohne Portierungsaufwand ablauffähig sind.

Durch den Einsatz massiv paralleler Systeme können außerdem wichtige Systemeigenschaften wie Verfügbarkeit, geringe Ausfallwahrscheinlichkeit, hohes Leistungspotential, Sicherheit und Verteilbarkeit erreicht werden.

SINIX und massiv parallele Architekturen

Um die Vorteile des Leistungspotentials dieser neuen Rechnergeneration ausnützen zu können, muß neue Software geschrieben oder bereits bestehende Software angepaßt werden, um die Parallelität ihrer internen Abläufe zu erhöhen. Meist können aber nur wenige Anwendungen von der Parallelisierung profitieren; der Großteil der Anwendungen benötigt keine parallele Verarbeitung. Portierungsaufwand für Anwendungen zu leisten, die diese neuen Mechanismen für ihre Ablauffähigkeit nicht benötigen, würde für viele Kunden eine überflüssige Investition bedeuten. Demgegenüber würde Siemens Nixdorf seinen Kunden die Vorteile dieser neuen Architekturen direkt verfügbar machen, indem das bestehende SINIX API weiterhin unterstützt und gleichzeitig ein erweitertes API für die explizite Unterstützung der Parallelverarbeitung angeboten wird.

Anwendungen stellen ständig höhere Anforderungen an die Leistungsfähigkeit der Rechner, auf denen sie ablaufen (Wachstumsfaktor 2 pro Jahr). Dadurch und durch die permanenten Innovationen im Hardware-Bereich (schnellere Prozessoren, größere Speicherkapazität) in Verbindung mit sinkenden Preisen, wird ein SINIX erforderlich, das Hardware-Architekturen nutzen kann, die im Umfang ständig wachsen. Der Bedarf an hoher Rechenleistung kommt vor allem aus den Bereichen des OLTP, großen Datenbank-Servern mit hoher Zugriffsfrequenz sowie entscheidungsorientierten Anwendungen. Derart leistungsfähige Rechner werden auch im kommerziellen Bereich neuartige Anwendungen ermöglichen, wie sie bisher nur im technisch-wissenschaftlichen Bereich bekannt waren (number crunching), z.B. Prognosemodelle für Börsenkurse, Zinsentwicklung, Schadensfälle usw.

Wie schon erwähnt wurde, ist die wichtigste Voraussetzung die Unterstützung des SINIX API. Wenn das bestehende SINIX-Betriebssystem auf massiv parallele Architekturen portiert werden soll, werden viele Erweiterungen nötig, um die Vorteile dieser Architekturen für den Anwender verfügbar zu machen. Zu diesem Zweck müssen zusätzliche Funktionen sowohl innerhalb als auch außerhalb des Systemkerns entwickelt oder erworben werden. Außerhalb des Systemkerns würde SINIX einen Server zur Verteilung der Arbeitsbelastung und ein Verwaltungswerkzeug wie das Distributed Management Environment von OSF benötigen, um problemlos ein komplexes, massiv paralleles System betreiben zu können. Ein solches System benötigt einen sogenannten Load Balancer, der bei der Prozessorzuweisung die gegenwärtige Auslastung der einzelnen Knoten berücksichtigt. Zusätzlich müssen Funktionen wie *remote_fork()* zur Verfügung stehen, d.h. Anwendungen, die die effektive Ausnutzung massiv paralleler Systeme ermöglichen. Wichtig ist dies vor

allem für die Entwicklung oder Konvertierung von Client-Server-Anwendungen. Daraus wird wiederum die Notwendigkeit ersichtlich, das Betriebssystem SINIX zu einer angemessenen Lösung für massiv parallele Rechner zu machen. Auf lange Sicht gesehen liegt die Lösung in einem mikrokernelbasierten SINIX, wie es im vorhergehenden Abschnitt besprochen wurde.

Ausblick

Multiprozessorrechner werden im UNIX-Umfeld in Zukunft an Bedeutung gewinnen. Multiprozessorrechner sind dabei, die obere Leistungsklasse mittlerer Systeme zu beherrschen. Verbesserter Systembus und ausgefeilte Cache-Architekturen sind wesentliche Voraussetzungen, um 20 oder 30 Prozessoren in einem System integrieren zu können. Bei Siemens Nixdorf befinden sich neue UNIX-Rechner mit bis zu 24 Prozessoren in der Planungs- und Entwicklungsphase. Daneben sind bereits Architekturen mit verteiltem Speicher und mehreren hundert Prozessoren untersucht worden. In naher Zukunft werden neue SINIX-Versionen an diese komplizierten Strukturen angepaßt werden und die Tür zu Mainframe-vergleichbarem Leistungspotential auf SINIX-Systemen öffnen.

Multimedia

Bisherige Anwendungen haben in der Regel nur Text, numerische und grafische Daten verarbeitet. Multimedia ermöglicht Wege zur Verarbeitung von digitalisierten Bildern, Ton und Videosequenzen. Multimedia setzt voraus, daß die verschiedenen Medien entweder in digitaler Form existieren, die vom Computer verarbeitet werden kann, oder daß eine analoge Form existiert, die über digitale Signale durch den Computer gesteuert werden kann. Multimedia setzt außerdem voraus, daß der Anwender sich mit dem Computer, auf dem alle oder eines dieser Medien Einsatz finden, im Dialog befindet, so daß eine natürlichere Mensch-Rechner-Schnittstelle entsteht.

Multimediadaten können die unterschiedlichsten Formate haben, sind erweiterbar, um neue Typen miteinzuschließen und können auf verschiedenste Art gespeichert, wiedergegeben oder erstellt werden. Dies macht Multimedia aus technischer Sicht zu einer besonderen Herausforderung. Um Multimedia-Eigenschaften in ein System zu integrieren, ist spezielle Hardware und/oder Software notwendig.

Multimedia verbindet nicht nur unterschiedliche Medien, sondern verschmilzt auch verschiedene Industriezweige miteinander, darunter:

- Kommunikation (Stimme und Daten)
- Computer und digitale Verarbeitung
- Verlagswesen und Werbung
- Bildung und Weiterbildung
- Nachrichten und Information
- Unterhaltung

Multimediaverarbeitung per Computer wird zur Zusammenarbeit von Computeranbietern und Firmen aus diesen Bereichen führen.

Multimedia-Anwendungen für Weiterbildung und Verkaufsförderung im kommerziellen Bereich bilden einen großen potentiellen Markt. 1991 gab es allein in den USA etwa 280.000 installierte Mulitmedia-Trainingssysteme. Die Vorstellung des Trainings „just in time" ist für Unternehmen in Bereichen, die sich schnell entwikkeln und starkem Wettbewerbsdruck ausgesetzt sind, von steigender Bedeutung. Multimedia-Trainingsprogramme werden eine größere Rolle bei der Deckung dieses Bedarfs spielen, da sie schnelle und effektive Möglichkeiten zur Übermittlung neuer Informationen bieten.

Anwender werden möglicherweise zuerst über dedizierte Systeme mit Multimediaverarbeitung bekannt, die Multimedia-Anwendungen in den Bereichen Verkauf, Information, Unterhaltung, Publishing oder Instruktion zum Ablauf bringen. Sie werden dann erwarten, daß auch von Universalcomputern der Zugriff auf Multimedia-Anwendungen möglich ist. Auch wenn Multimedia den Markteinstieg in den oben erwähnten Bereichen der dedizierten Systeme macht, wird bald auch der große Markt kommerzieller Computeranwendungen um Multimedia nicht herumkommen. Für viele technische Hürden, die die Marktdurchdringung durch Multimedia in der Vergangenheit behinderten, gibt es mittlerweile Lösungen in einem akzeptablen Preis-/Leistungsverhältnis und macht diese für den Bereich kommerzieller Anwendungen geeignet.

Notwendige Infrastruktur

Multimedia-Anwendungen erfordern eine effiziente Infrastruktur mit unterstützenden Technologien wie Autorensystemen, Multimedia-Bibliotheken, Netzwerken und Netzwerkschnittstellen mit ausreichender Bandbreite und Echtzeit-Unterstützung. Multimedia muß auf offene Art und Weise mit Standards entwickelt werden, die Portabilität und Interoperabilität sicherstellen. Schließlich muß Multimedia effektiv in den Bereich kommerzieller Anwendungen integriert werden.

Die Entwicklung von Multimedia-Anwendungssoftware ist ohne ein hochfunktionales Autorensystem nicht praktikabel. Die Medien, die in einer typischen Multimedia-Anwendung zum Einsatz kommen (digitaler Ton, Animation, Grafik, überlagertes Bild, usw.) sind zu umfangreich, um auf einer niedrigeren Ebene als einem Autorensystem gesteuert oder programmiert zu werden. Um eine einzige Bildschirmseite einer gängigen Selbstbedienungs- oder CBT-Anwendung (**C**omputer **B**ased **T**raining) herzustellen, würden tausende von Zeilen C-Quellcode benötigt. Dieses Thema erledigt sich häufig von selbst, da in den meisten Fällen derjenige,

der die größte Erfahrung im Bereich der Multimedia-Anwendungen hat, nur über geringe Programmiererfahrung verfügt. Selbst wenn man einen erfahrenen Multimedia-Entwickler mit einem Experten auf dem Gebiet der Anwendungsprogrammierung zusammenbringen würde, wäre das Ergebnis weder kosteneffektiv noch termingerecht. Das typische Entwicklungsszenarium einer Mulitmedia-Anwendung ist folgendes: ein Experte für Anwendungsentwicklung arbeitet mit einem auf Autorensystemen erfahrenen Ingenieur zusammen.

Erforderlich sind zudem Bibliotheken von Multimedia-Bausteinen, um Zeit und Kosten für die Entwicklung einer Multimedia-Anwendung zu reduzieren. Ziel von Siemens Nixdorf ist es, eine konsistente verteilte Multimedia-Architektur mit wiederverwendbaren, anspruchsvollen Bausteinen an der Spitze multipler spezifischer Multimedia-Bibliotheken sowohl für PC- als auch für SINIX-Plattformen zur Verfügung zu stellen. Die Architektur wird sich an einem Schichtenmodell orientieren, so daß Entwickler die entsprechende Ebene finden. Zusätzlich werden Bausteine mit anspruchsvollen prozeduralen APIs für Programmierer zur Verfügung gestellt werden. Zu den geplanten Bausteinen gehören:

- Ton-Baustein
- Telefonschnittstellen-Baustein
- Fax-Baustein
- Video-Baustein
- Zusätzliche Bausteine (Kommentardienst, OCR-Dienst, Service für einen virtuellen Schalter)

Der Einsatz von Multimedia bringt mit sich, daß zukünftige Datenverarbeitungssysteme in der Lage sein müssen, weitaus größere Datenmengen und weitaus höhere Datendurchsätze sowohl innerhalb eines Systems als auch im Netzverbund mehrerer Systeme zu bewältigen. Betroffene Systemkomponenten sind u.a. Festplatten- und Hauptspeicher, Datenbanken, Echtzeitverarbeitung, Ein-/Ausgabe-Raten sowie Netzübertragungsraten. Multimedia stellt bezüglich der Echtzeitverarbeitung hohe Anforderungen an die Kommunikationsinfrastruktur. LANs wie Ethernet, Token Ring und FDDI genügen diesen Anforderungen nicht. Der Transfermodul ATM bietet jedoch die technischen Voraussetzungen sowohl für den lokalen als auch für den Weitverkehrsverbund. Massiv parallele Systeme aus Hunderten von Prozessoren mit extrem schnellen proprietären Verbindungen werden einen unteren Durchsatz in der Größenordnung von 1,4 Gigabits/Sekunde erbringen.

Ein weiteres wichtiges Thema bezüglich Multimedia-Daten ist das der Standards. Ohne existierende Standards ist es für Anwender ausgesprochen schwierig, Daten auszutauschen. Dieses Problem wird akut, sobald Multimedia auf einer Mulituser-Netzwerkumgebung eingesetzt wird. In jedem dieser wichtigen Bereiche laufen Standardisierungsbestrebungen. Daneben werden Standards für die Kompression von digitalisierten Bildern notwendig. Bestimmte Anwendungen können dabei Qualitätseinbußen hinnehmen, andere nicht (z.B. Röntgenbilder). Neue Techniken der Speicherung von Videodaten auf neuartigen Medien werden erforderlich, um auch hier einen kontinuierlichen Datenfluß zu gewährleisten. Es existieren bereits eine Reihe von Organisationen, die Standards entwickeln, die die Struktur von Multimediasystemen in der Zukunft bestimmen werden (z.B. Standards wie MPEG, MCI, CD ROM XA usw.).

Eine der großen Herausforderungen von Multimedia ist, daß hier eine konsistente Integration der unterschiedlichen Medien gefragt ist. Es genügt nicht, einfach Multimedia-Daten zur Anzeige zu bringen, dies demonstriert lediglich Technologie, bietet aber keine wirtschaftlich interessanten Lösungen. Einige Beispiele integrierter Werkzeuge wären heutige Autorensysteme und die entsprechenden Abspielgeräte, Mischdokumente und Hyper-Medien. Mit diesen Werkzeugen beginnt das eigentliche Potential von Multimedia sichtbar zu werden. Von größtem Interesse sind zudem Anwendungen wie Videokonferenzen (Desktop-Konferenz) mit der Möglichkeit, gemeinsam an einem Dokument oder einer Grafik arbeiten zu können (Joint-Editing).

Multimedia-Aktivitäten von Siemens Nixdorf

Überblick

Es ist Ziel von Siemens Nixdorf, auf allen zukünftigen Systemen Empfang, Übertragung, Verarbeitung und Wiedergabe von Mulimedia-Daten zu ermöglichen. Siemens Nixdorf beabsichtigt, sich mit Partnern zusammenzuschließen, die im Multimedia-Bereich etabliert sind, um Technologien auszutauschen und die Verfügbarkeit von Multimedia-Anwendungen auf einer breiten Palette von Siemens Nixdorf Hardware-Plattformen zu beschleunigen.

Da entsprechende Standards ein Muß für den Erfolg von Multimedia sind, wird Siemens Nixdorf Standardisierungsbestrebungen beobachten und unterstützen, um sicherzustellen, daß die Einführung proprietärer Schnittstellen und Formate vermieden wird. Siemens Nixdorf wird das SINIX API erweitern, um Multimedia konform mit Standard-UNIX-Erweiterungen zu behandeln. Hierzu werden bestehende und De-facto-Standards ebenso wie neu entstehende Standards Beachtung finden.

Siemens Nixdorf wird offene Multimedia-Lösungen mit dem Ziel anbieten, diese in heterogenen Umgebungen zu verbreiten. Während das derzeitige Interesse des Multimedia-Marktes sich vor allem auf PCs beschränkt, werden SINIX-Systeme mit ihrer umfangreicheren Netzfähigkeit und dem Multiuser-/Multitasking-Betriebssystem anspruchsvollere Lösungen ermöglichen. Siemens Nixdorf verfolgt verteilte Multimedia-Lösungen, die die nahtlose Integration von PC- und SINIX-Umgebungen ermöglichen.

Mercury: Multimedia-Dokumenten-Austausch

Mercury ist ein SINIX-basiertes Multimedia-Austauschsystem für Mischdokumente, das von Siemens Nixdorf entwickelt wird. Mercury fällt in die Kategorie der „Groupware"-Produkte. Das System existiert derzeit als Prototyp und wird schnellstens zu einem Produkt entwickelt. Die wesentlichen Eigenschaften von Mercury sind:

- Die Einbindung multipler Datentypen in eine einzige zusammengesetzte Nachricht für elektronische Post (E-Mail).
- Die menschliche Stimme als zusätzlicher Datentyp, der für eine Nachricht über elektronische Post (E-mail) zur Verfügung steht.
- Die Fähigkeit, ein Fax als Teil einer elektronischen Nachricht zu empfangen.
- Die Fähigkeit, ein Fax von Mercury aus zu senden, indem im Adreßfeld der Nachricht eine Fax-Nummer angegeben wird.
- Eine Kommentarmöglichkeit, die dem Anwender erlaubt, gesprochene Kommentare an Dokumente anzufügen (Sprachannotation).
- Eine Telefonschnittstelle, die die Weiterleitung elektronischer Post und elektronischer Faxe an ein beliebiges Faxgerät ermöglicht.
- Die Nutzung des vorhandenen Telefonnetzes als Sprach-Ein- und Ausgabegerät.

Die Multimedia-Architektur für SINIX wird von Anfang an die Voraussetzungen beachten, die für eine erfolgreiche Integration von Multimedia in eine Netzwerk-Umgebung notwendig sind:

- Client-Server-Struktur
- Netzwerkunterstützung mit hoher Bandbreite
- Unterstützung verteilter, objektorientierter Datenbasen

Objektorientierung

Anwendungen werden in zunehmendem Maß objektorientiert gestaltet. Sie werden auf feinstrukturierte, wiederverwendbare Bausteine zurückgreifen, die Querverweise, Bindefähigkeit und Zugangstransparenz ermöglichen. Solche Anwendungen werden einfacher zu entwickeln, zu warten und zu verbessern sein.

Das Konzept der Objektorientierung ist seit über einem Jahrzehnt im Gespräch und in der Erprobung. Während dieser Zeit wurden die Architektur objektorientierter Software (Analyse, Design und Programmierung) sowie objektorientierte Bedienoberflächen entwickelt. Das volle Potential der Objektorientierung ist jedoch noch lange nicht ausgeschöpft. Objektorientierung wird in den nächsten zehn Jahren die führende Softwaretechnologie darstellen.

Benachbarte Themenbereiche sind die Integration objektorientierter Technologien in eine verteilte, heterogene Umgebung sowie der Standardisierungsprozeß.

Objektorientierte Software-Architektur

Vorrangiges Ziel der objektorientierten Software-Entwicklung ist die Senkung der Entwicklungs- und Wartungskosten, indem möglichst viele Probleme im Stadium der Definition von Objekten und deren Methoden gelöst werden und nicht erst zu einem späteren Zeitpunkt, wo wesentlich höhere Kosten anfallen. Daneben verringert ein vertikal integrierter, objektorientierter Ansatz aufgrund der Klarheit seines Designs und des Objektzugriffs nur über seine Methoden die Fehlerwahrscheinlichkeit in späteren Phasen.

Eine Studie des amerikanischen Verteidigungsministeriums [Hughes DoD Composite Software Error History] zur Fehlerhäufigkeit in Software-Entwicklungsprojekten veröffentlichte die folgende Fehlerstatistik für die verschiedenen Phasen des Software-Entwicklungsprozesses:

Entwicklungsphase	Kosten	Vorhandene Fehler	Gefundene Fehler
Anforderungsstudie	5%	55%	18%
Design	25%	30%	10%
Codierung und Komponententest	10%		
Integration und Test	50%	10%	50%
Validierung und Dokumentation	10%		
Betrieb/Wartung		5%	22%

Diese Studie bezog die anteiligen Projektkosten in der Betriebs- und Wartungsphase nicht mit ein. Zu diesem Zeitpunkt fällt ein Software-Produkt häufig in die Verantwortung einer anderen Organisation und die Kosten sind entweder nicht feststellbar, die Zahlen nicht verfügbar oder aber sie werden mit den Betriebs- und Wartungskosten anderer Software verrechnet. Es ist unbestreitbar, daß die Behebung eines Fehlers umso teurer ist, je weiter die Projektphase, in der er entdeckt wird.

Eine Untersuchung des Marktforschungsinstituts Butler/Bloor setzt für die Fehlerbehebung in den einzelnen Phasen eines Software-Projekts die folgenden Kosten an:

Anforderungsstudie	3%
Design	9%
Programmierung	7%
Test	15%
Wartung	67%

Der Software-Entwicklungsprozeß sieht sich einem grundlegenden Dilemma gegenüber: Die Komplexität der zu entwickelnden Software-Systeme wächst laufend, demgegenüber stehen die grundlegenden Einschränkungen unserer Fähigkeit, diese Komplexität zu beherrschen. Eine Lösung dieses Problems liegt darin, im Software-Entwicklungsprozeß objektorientierte Analyse-, Design- und Programmiermethoden anzuwenden.

Objektorientierte Analyse „ist eine Analysemethode, die die Anforderungen aus der Sicht der Klassen und Objekte untersucht, die in der Terminologie des Problembereichs auftauchen" (... "is a method of analysis that examines requirements from the perspective of the classes and objects found in the vocabulary of the problem domain.") [nach: Object-Oriented Design, Grady Booch]. Das heißt, daß die Analyse der Anforderungen an ein Produkt so erfolgt, daß sie durch die Objekte und Interaktionen der Objekte ausgedrückt werden kann.

Objektorientiertes Design „ist eine Designmethode, bestehend aus dem Prozeß der objektorientierten Strukturierung und einer Notation sowohl für logische und physische als auch statische und dynamische Modelle des Systems, das entworfen wird" (... "is a method of design encompassing the process of object-oriented decomposition and a notation for depicting both logical and physical as well as static and dynamic models of the system under design.") [nach: Object-Oriented Design, Grady Booch]. Objektorientierte Strukturierung verwendet Klassen- und Objektabstraktionen, um ein System logisch zu strukturieren, im Gegensatz zum strukturierten Design, das die algorithmische Abstraktion benutzt. Objekte können als Datenabstraktion mit einer Schnittstelle benannter Operationen und einem verborgenen lokalen Zustand definiert werden, während Klassen Typen sind, die mit Objekten verbunden sind.

Unter objektorientierter Programmierung versteht man „eine Methode der Implementierung, in der Programme als kooperative Sammlungen von Objekten organisiert sind, von denen jedes einzelne einen Vertreter einer bestimmten Klasse darstellt, und die Klassen alle Bestandteil einer Klassenhierarchie sind, die über Verwandtschaftsbeziehungen verbunden sind" (... "is a method of implementation in which programs are organized as cooperative collections of objects, each of which represents an instance of some class, and whose classes are all members of a hierarchy of classes united via inheritance relationships.") [nach: Object-Oriented Design, Grady Booch].

Entwickler arbeiten bereits mit objektorientierten Programmiersprachen und Werkzeugen, die auf lange Sicht gesehen die Programmiertechniken der Zukunft darstellen. Während Sprachen wie Smalltalk, die von Anfang an objektorientiert entwikkelt wurden, weiter in Verwendung bleiben werden, scheint es, daß Sprachen wie C++, die eine objektorientierte Schicht über einer traditionellen Sprache darstellen, favorisiert werden. Objektorientierte Programmierung bietet eine Reihe von Vorteilen, die sich hauptsächlich in zwei Kategorien einteilen lassen: Einschalung und Wiederverwendbarkeit. Einschalung ermöglicht den Datenzugriff lediglich auf die Art, wie er von der Software, die das Objekt implementiert, zugelassen wird.

Die Entwicklung von Anwendungen in wirklich modularer Struktur wird dadurch unterstützt, daß unbeabsichtigte Beeinflussungen bestehender Objekte unterbunden werden. Zudem wird die inkrementelle Anwendungsentwicklung erleichtert, da während des Entwicklungsprozesses Fehlerfreiheit aufrechterhalten wird. Wenn die Standardisierung der Schnittstellen zwischen unabhängig entwickelten Anwendungen und Anwendungskomponenten erreicht ist — ein Charakteristikum objektorientierter Umgebungen — dann werden Kosten und „Time-to-Market" eingespart werden können, indem bereits bestehende Realisierungen von Objektklassen verwendet werden.

Der Übergang zu objektorientierten Technologien erfordert, daß bestehende Anwendungen neben einer objektorientierten Umgebung existieren können. Sie werden in die objektorientierte Umgebung mit einem Integrationsgrad eingebettet werden müssen, der an die Charakteristika der jeweiligen Anwendung angepaßt ist.

Objektorientierte Bedienoberflächen

Die Entwicklung von Bedienoberflächen ist einer der ersten Bereiche, in denen objektorientierte Ansätze erfolgreich auf breiter Basis verwirklicht wurden. Drag-and-Drop-Funktionen, die auf Icons basieren (vgl. SINIX/windows) verdeutlichen die Errungenschaften objektorientierter Realisierungen. Objektorientierte Bedienoberflächen weisen gegenüber traditionellen Bedienoberflächen viele Vorteile auf. Sie verfügen sowohl über eine umfassendere Sichtweise der Benutzerumgebung, als auch über einen intuitiveren Ansatz, um Aktionen zu initiieren. Objekte werden dem Benutzer durch Symbole präsentiert (z.B. Icons), die auf vergleichbare Weise manipuliert werden können wie die Objekte der realen Welt. Seit Jahren gibt es Bedienoberflächen, die einen objektorientierten Ansatz zumindest teilweise verwirklichen, z.B. Apple MacIntosh und New Wave von Hewlett Packard. Fortgeschrittenere Beispiele finden sich im CAD-Bereich (**C**omputer **A**ided **D**esign), wo den Anwendern eine Reihe von Objekten zur Verfügung steht, die funktionale Komponenten repräsentieren. Der Anwender ist in der Lage, diese Objekte auf sehr realistische Weise zu manipulieren und zu verbinden. Ein konsistenter objektorientierter Ansatz für Bedienoberflächen kann den unterschiedlichsten Anwendungen ein einheitliches „Look and Feel" geben, deren Anwenderfreundlichkeit erhöhen und gleichzeitig den notwendigen Lernaufwand reduzieren.

Objektorientierte verteilte Verarbeitung

Eine der häufigsten Anforderungen, die Anwender heute an Computeranbieter stellen, ist die der Interoperabilität der Systeme. Damit objektorientierte Anwendungen in einer heterogenen Umgebung funktionieren können, muß eine Spezifikation entwickelt werden, die beschreibt, wie ferne Objekte miteinander kommunizieren können. Im September 1991 hat die OMG (**O**bject **M**anagement **G**roup) eine Standard-Schnittstelle für den ORB (**O**bject **R**equest **B**roker) ausgewählt, die auf einem gemeinsamen Vorschlag von DEC, HP, HyperDesk, NCR, Object Design und SunSoft basierte, die sogenannte CORBA (**C**ommon **O**bject **R**equest **B**roker **A**rchitecture). Details dazu sind im OMG-Dokument „The Common Object Request Broker: Architecture and Specifications" enthalten.

Die OMG arbeitet derzeit daran, diese Spezifikation in den Bereichen der ORB-Interoperabilität, des Objektsharings, des Object Repository und der Konformitätsprüfung auszubauen. Der **O**bject **R**equest **B**roker (ORB) stellt Mechanismen zur Verfügung, mit denen Objekte auf transparente Weise Anfragen senden und Antworten empfangen. Dabei gewährleistet der ORB Interoperabilität zwischen Anwendungen auf verschiedenen Rechnern in heterogenen verteilten Umgebungen und die nahtlose Verbindung multipler Objektsysteme. Der ORB deckt dabei den Zugriff auf Namen, die Übertragung, Synchronisierung, Aktivierung und Deaktivierung von Objekten, die Ausnahmebearbeitung und die Sicherheit ab.

Die Open Software Foundation hat CORBA zur Grundlage für das OSF/DME gemacht. DME wird einen Management Request Broker zur Verfügung stellen, der die OSF-Realisierung einer Obermenge des CORBA-Standards sein wird. Noch ungelöst ist die Frage, wie der ORB mit einer Remote-Procedure-Call-Implementierung (wie etwa der RPC des DCE) integriert werden kann. Im wesentlichen geben die ORBs die feinstrukturierte Überwachung von Netzwerkoperationen zugunsten der Flexibilität in der Organisation der Kommunikation zwischen den Einheiten auf (Vererbung, multiple Kommunikationsmodelle, ereignisorientierter Programmierstil). Dies bedeutet jedoch kein großes Opfer: ORBs bieten im Grunde alle Vorteile der RPCs und darüberhinaus Wiederverwendbarkeit und der Flexibilität objektorientierter Software. Das richtige Modell wäre möglicherweise dasjenige, welches einen ORB auf OSF/DCE aufsetzen läßt. Siemens Nixdorf wird die Aktivitäten der OSF und der OMG insbesondere in diesem Bereich verfolgen und daran arbeiten, beide Organisationen dahingehend zu beeinflussen, beide Ansätze zu vereinen.

Standardisierung

Die Object Management Group (OMG) ist die führende Organisation, die an der Standardisierung objektorientierter Konzepte und Technologien arbeitet. Siemens Nixdorf war nicht von Anfang an Mitglied der OMG, hat jedoch einen nicht unbeträchtlichen Beitrag zur Entwicklung der ersten Version des OMA Guides geleistet. Dies ermöglicht Siemens Nixdorf, am Ball zu bleiben und den Standardisierungsprozeß zu beeinflussen, um vorbereitet zu sein, wenn diese Technologien Produktreife erlangen.

In den folgenden Bereichen haben Standardisierungsbestrebungen bereits begonnen: Referenzarchitektur, Objektmodell, Kommunikation zwischen Anwendungen (z.B. Object Request Broker,) Verwaltung von Objekten über ihren gesamten Lebenszyklus (Objektdienste für Erstellung, Löschen, Objektzugriff und Objektlokalisierung), generische Anwendungsfunktionen (gemeinsame Mechanismen für z.B. Druck, Datenbankzugriff und elektronische Post), Erstellung neuer Objektklassen durch Modifikation bestehender Klassen (Generalisierung oder Spezialisierung bestehender Klassen, wie durch die Objektdienste zur Verfügung gestellt). Das Ziel der OMG ist es nach eigenen Aussagen, „die bestehende Komplexität zu reduzieren, geringere Kosten zu ermöglichen und die Einführung neuer Anwendungen zu beschleunigen. Die OMG beabsichtigt, dies durch die Einführung eines architektonischen Rahmens mit zugehörigen detaillierten Schnittstellenspezifikationen zu verwirklichen. Diese Spezifikationen werden die Industrie hin zu interoperablen, wiederverwendbaren, portablen Software-Komponenten führen, die auf objektorientierten Standardschnittstellen basieren."

X/Open hat CORBA zugestimmt und als entsprechende vorläufige Spezifikation veröffentlicht, gemäß der X/Open-Strategie, nützliche Standards anderer Organisationen anzuerkennen und zu unterstützen. Der ORB wird in das X/Open Common Applications Environment aufgenommen werden. X/Open und UNIX International (UI) finanzieren gemeinsam die Entwicklung eines Testwerkzeugs zur Validierung von ORB-Implementationen.

Anhang A Standards für offene Systeme

Überblick

Standards für offene Systeme entstammen vor allem drei Quellen:

- Standardisierungsgremien
- Konsortien
- Produkten die zum Industriestandard wurden.

Zu den Standardisierungsgremien gehören ISO, IEEE und ANSI in den USA und DIN in Deutschland. Beispiele für Konsortien sind X/Open und die Open Software Foundation. Ein Produkt, das zum Industriestandard wurde, ist beispielsweise NFS.

Vielleicht würden in einer idealen Welt alle Standards von Standardisierungsgremien herausgegeben. Sobald sie veröffentlicht wären, hätte jeder die gleiche Möglichkeit, diese Standards in Produkte zu integrieren. Die Entwicklung formaler Standards ist jedoch ein sehr langsamer Prozeß. Entwürfe müssen geschrieben werden, langwierige Reviewverfahren durchgeführt und Themen kontrovers diskutiert und verhandelt werden. Wenn die offenen Systeme einzig und allein vom schneckengleichen Fortschritt der Standardisierungsgremien abhängig wären, hätten sie keine praktische Lösung zu bieten. Zudem bewegen sich formale Standards oft fern der aktuellen Praxis, so daß ihre Verwirklichung problematisch und langsam sein kann.

Konsortien haben sich als effektive zusätzliche Quelle für die Standards offener Systeme erwiesen. Die bekanntesten Konsortien sind X/Open und OSF. Das Konzept von X/Open ist die Veröffentlichung schriftlicher Standard-Spezifikationen, während OSF seine Standards in Form der Bereitstellung von Sourcelizenzen realisiert. In Konsortien findet eine höchst produktive Zusammenarbeit von Anbietern mit dem Ziel statt, umfassende Standards so schnell als möglich herauszubringen und in Produkten zu implementieren. Natürlich müssen Konsortien sowohl einer Eigenkontrolle als auch der Regierungskontrolle unterliegen, um zu verhindern, daß sie

sich zu Kartellen entwickeln, die Innovationen eher unterdrücken würden. Im Fall des Ansatzes von X/Open müssen die vorgeschlagenen Standards technisch gut ausgereift sein und weite Akzeptanz finden. Das OSF-Konzept setzt voraus, daß die Sourcecodes auf vielen Systemen implementierbar sind und zu einem vernünftigen Preis zu allgemein anerkannten Lizenzierungsbedingungen angeboten werden.

Standards für offene Systeme, die auf einem Produkt basieren, das zur Industrienorm wurde, sind alles andere als ideal, da sie zumindest anfangs von einem einzigen Unternehmen produziert und überwacht werden. Trotzdem können sie für Anbieter und auch Anwender von großem Wert sein, wenn solche Unternehmen in der Lage sind, Schnittstellenspezifikationen zu veröffentlichen und/oder Sourcecodelizenzen zu angemessenen Preisen und Bedingungen anzubieten.

Standardisierungsgremien

ISO

Die höchste Autorität für Standards in der Informationstechnologie in der Welt, das JTC1 (**J**oint **T**echnical **C**ommittee), ist ein Zusammenschluß der ISO (**I**nternational **O**rganization for **S**tandardization) und des IEC (**I**nternational **E**lectrotechnical **C**ommittee). Die Mitglieder der ISO sind die nationalen Standardisierungsgremien wie ANSI (USA) und BSI (Großbritannien), was die ISO zu einer mächtigen Organisation werden ließ. Verschiedene Gruppierungen arbeiten eng mit der ISO zusammen, darunter das IEEE (**I**nstitute of **E**lectrical and **E**lectronic **E**ngineers) und die ECMA (**E**uropean **C**omputer **M**anufacturer's **A**ssociation). Im Bereich der Computerkommunikation bestens durch ihr OSI 7-Schichtenmodell, ihre Protokollstandards und Kommunikationssprofile bekannt, entwirft die ISO Dokumente für internationale Standards in unterschiedlichen Bereichen, die u.a. die Telekommunikation und Netzwerke betreffen. Andere gemeinsame Standards des CCITT (**C**onsulting **C**ommittee on **I**nternational **T**elegraphy and **T**elephony), bekannt durch seine CCITT-Nummern, und ISO/IEC schließen X.25 (Netzwerkdienste), X.400 (elektronische Nachrichtendienste) und X.500 (Directory Service) ein.

In den USA wurden diese Standards schließlich in einem Anforderungskatalog der Regierung, dem U.S. Government OSI Profile (GOSIP), und später in einem Standard, dem FIPS (**F**ederal **I**nformation **P**rocessing **S**tandard), zusammengefaßt, der für Bundesbehörden Gesetzeskraft hat. Für kommerzielle Anwender ist jedoch keiner von beiden rechtlich verbindlich. In Großbritannien wurden einige dieser Standards in das öffentliche Recht integriert.

IEEE

Die IEEE ist eine Berufsvereinigung von Ingenieuren, die sowohl für Hardware als auch für Software Standards entwickelt. Im Bereich der offenen Systeme ist die IEEE vor allem durch POSIX (**P**ortable **O**perating **S**ystem Standard) bekannt, einem Standard, der die Schnittstelle zwischen dem Kern des Betriebssystems und Anwendungsprogrammen beschreibt. IEEE ist vom ANSI zur POSIX-Standardisierung akkreditiert und tritt im ISO/IEC JTC1 als Zulieferer für die POSIX-Standardisierung auf.

ANSI

ANSI (**A**merican **N**ational **S**tandards **I**nstitute) vertritt die USA in der ISO. In den USA entwickelt ANSI nationale Standards für die unterschiedlichsten Bereiche. Am besten bekannt sind wohl die ANSI-Standards für Programmiersprachen wie etwa ANSI COBOL und ANSI C.

Konsortien

Zusätzlich zu X/Open und OSF schossen Konsortien aus dem Boden, um De-facto-Standards für unterschiedlichste Aspekte offener Systeme zu koordinieren und zu entwickeln. Die Liste dieser Konsortien umfaßt z.B. das MIT X Konsortium, das **N**etwork **M**anagement **F**orum (NMF), die **O**bject **M**anagement **G**roup (OMG), das SIGMA-Konsortium, die **S**tandards **P**romotion and **A**pplication **G**roup (SPAG), die SQL Access Group, die X.400 API Association, die Corporation for Open Systems (COS), das ODA-Konsortium, die Petrotechnical Open Software Corporation (POSC), das Unicode-Consortium und UNIX International (UI).

X/Open

X/Open ist das bedeutendste Konsortium im Umfeld der offenen Systeme. Ursprünglich wurde X/Open durch eine Gruppe europäischer Computerhersteller gebildet. Diese Basis wurde jedoch erweitert, um auch amerikanische und japanische Firmen zu beteiligen. Daneben sind zwei wichtige Konsortien offener Systeme, die OSF (die **O**pen **S**oftware **F**oundation) und UI (**U**NIX **I**nternational), Mitglieder von X/Open. X/Open selbst ist kein Standardisierungsgremium, wählt jedoch Standards der Standardisierungsgremien wie beispielsweise den POSIX-Standard der IEEE oder OSF/DCE aus und übernimmt diese als Teil des **X**/Open **P**ortability **G**uide (XPG).

X/Open hat folgende Zielsetzungen: Portabilität von Anwendungen auf der Ebene der Sourcen, portierbare Netzwerkdienste, um Anwendungen, die auf verschiedenen Rechnern ablaufen, über ein Netzwerk miteinander zu verbinden sowie einen konsistenten Ansatz im Bereich der Bedienoberflächen.

Nach der dritten Version des XPG (XPG3) begann X/Open nach und nach mit verschiedenen Konsortien Kooperationsverträge abzuschließen, um die Arbeit dieser Konsortien mit der Entwicklung der X/Open-Spezifikationen zu koordinieren und die Ergebnisse als X/Open -Spezifikationen zu veröffentlichen. Es genügt also, die Aktivitäten von X/Open zu verfolgen, wenn man sich über die Aktivitäten der wichtigsten Standardisierungsgremien und Konsortien informieren möchte.

Die wichtigsten und bekanntesten Ergebnisse dieser Kooperationsbestrebungen sind die X/Open-Spezifikationen für das X Window System, der Object Request Broker, SQL, der Remote Data Base Access, das OSI Directory API und die E-Mail APIs. Die Veröffentlichung weiterer Spezifikationen wird folgen, darunter die X/Open DCE-Spezifikationen.

XPG3

X/Open veröffentlichte 1985 den XPG1. 1989 wurde mit der Veröffentlichung des XPG3, der mit einem Test- und Warenzeichensystem gekoppelt ist, ein Durchbruch erzielt. Alle großen Hersteller boten XPG3-konforme Systeme an und Kunden begannen, die XPG3-Konformität, zertifiziert durch das X/Open-Warenzeichen, zu verlangen. Der XPG3 besteht aus sieben Bänden, die folgende Themen behandeln:

- Systemkommandos und Funktionen
- Systemschnittstellen und Header
- Internationalisierung
- Terminalschnittstellen
- Interprozeßkommunikation
- Sourcecode Transfer
- Programmiersprachen
- Datenverwaltung
- Fensterverwaltung
- Netzwerkdienste

XPG4

Die Veröffentlichung des XPG4 im Oktober 1992 enthielt 25 neue oder überarbeitete Spezifikationen, zusammen mit offiziellen Standards für 22 Komponenten, für die das X/Open-Warenzeichen beantragt werden kann. Daneben wurden die Komponenten zu sogenannten Profilen zusammengefaßt, die die Integration der dazu konformen Produktkonfigurationen sicherstellen und für die es ebenfalls X/Open-Warenzeichen gibt.

Open Software Foundation

Die Open Software Foundation (OSF) wurde 1988 von sieben bedeutenden Computerherstellern gegründet: Apollo, Bull, DEC, HP, IBM, Nixdorf und Siemens. Zu diesem Zeitpunkt fürchteten diese Unternehmen, daß sich eine Zusammenarbeit von AT&T, dem Urheber des Betriebssystems UNIX mit Sun Microsystems sich gegen die Ziele der offenen Systeme auswirken würde. Ihre Absicht war, OSF Entwicklungstechnologien anzubieten, die nicht von einer einzelnen Firma kontrolliert wurden und diese zu einem angemessenen Preis als Sourceprodukte zu lizensieren. Auf diese Art können viele Anbieter aus den Sourcen Produkte entwickeln und OSF konnte einen De-facto-Industriestandard setzen. Die OSF besteht derzeit aus über 200 Mitgliedern, darunter Computeranbieter, unabhängige Softwarehäuser, Hersteller von Halbleitern, andere Konsortien, Bildungseinrichtungen und Behörden. In den ersten fünf Jahren ihres Bestehens hat die OSF bereits das Betriebssystem OSF/1, die grafische Bedienoberfläche Motif und DCE, das **D**istributed **C**omputing **E**nvironment, vorgelegt.

Standards für das Netzmanagement

Die Spezifikationen, die für die Standardisierung der Netzwerkverwaltung benötigt werden, fallen in das Aufgabengebiet der Arbeitsgruppe ISO/IEC JTC1 SC21 WG4. Diese Spezifikationen wurden auch vom CCITT (**C**onsulting **C**ommittee on **I**nternational **T**elegraphy and **T**elephony) übernommen. Das OSI-Managementssystem wurde unter Zuhilfenahme des OSI Referenzmodells (ISO 7498) definiert. Die Beschreibung dieses Systems setzt den Umfang der Grundlagen für OSI-Management und -Architekturen fest, definiert die Funktionsbereiche des OSI-Management und trifft eine Unterscheidung zwischen System- und Netzverwaltung. Die übrigen ISO-Dokumente zur Netzwerkverwaltung umfassen Verwaltungsfunktionen, Verwaltungsdienste (z.B. CMIS, **C**ommon **M**anagement **I**nformation **S**ervices) zum Austausch von Verwaltungsinformationen und die Struktur der Verwaltungsinformationen. Kurz gesagt, sie beinhalten sowohl das Informationsmodell als auch Regeln zur Definition der Objekte, die gesteuert werden sollen sowie deren Attribute.

Weitere Spezifikationen für die Verwaltung offener Systeme entstanden im Umkreis der Internet Organisation. Diese setzt De-facto-Standards für die Steuerung und Überwachung von TCP/IP-Netzwerken, die im Rahmen von Internet-Spezifikationen, sogenannten RFCs (**R**equests **f**or **C**omments), festgelegt wurden.

Diese Standardisierungsbestrebungen werden unterstützt durch ISO, CCITT, die Internet Community, OSF, X/OPEN, regionale Workshops für offene Systeme (in Europa: der EWOS, **E**uropean **W**orkshop for **O**pen **S**ystems) sowie durch eine Organisation von Anbietern und Telekommunikationsbehörden, die den Namen NM Forum (**N**etwork **M**anagement **F**orum) trägt. Unter anderem gehört die Spezifikation eines gemeinsamen Objektkatalogs zu diesen Bestrebungen. Siemens (und ebenso Siemens Nixdorf) trat dem NM-Forum bereits im Mai 1990 als aktives Mitglied bei.

Sicherheitsstandards

In der POSIX-Arbeitsgruppe 1003.6 werden Schnittstellenbeschreibungen zur Protokollierung, Zugangskontrolle und den Privilegien seit längerer Zeit diskutiert. Man kann davon ausgehen, daß diese Spezifikationen auch von X/Open und der JTC1 SC22 anerkannt werden, sobald eine Einigung erreicht wird. X/Open hat bereits einen *Leitfaden zur Sicherheit* herausgegeben, der Unterstützung beim sicheren Einsatz offener Systeme bietet. Eine Spezifikation der Protokoll- und Paßwortverwaltung wurde in Form eines „Snapshot" herausgegeben. Bei X/Open wurde zudem eine Empfehlung für ein verteiltes Sicherheitskonzept eingereicht, das auf asymmetrischen Verschlüsselungsverfahren basiert.

Aufgrund der großen Bedeutung der TCP/IP-Protokolle im UNIX-Umfeld müssen die Internet-Spezifikationen (RFCs) in die Erwägungen miteinbezogen werden. RFC 1108 legt die Optionen für IP-Header für den Transport von Sicherheitslabels im Internet fest.

Im Rahmen der ISO befassen sich eine Reihe von Arbeitsgruppen mit Fragen der Sicherheit. Die nachstehende Tabelle gibt eine Übersicht über Gruppen und Themen.

ISO/IEC JTC1	
SC6	OSI Netzwerk und Transportschichten
SC14	EDI Sicherheit
SC17	Identifizierungs- und Kreditkartensicherheit
SC18	Sicherheit von Text- und Bürosystemen
SC21	OSI Informationsvermittlung, Transfer und Verwaltung
SC22	Betriebssystemsicherheit (POSIX)
SC27	IT-Sicherheitstechniken Anforderungen für IT-Sicherheitsdienste Sicherheitsrichtlinien
ISO/TC46	Sicherheit von Informationssystemen
ISO/TC65	sicherheitskritische Systeme
ISO/TC68	Bankensicherheit
ISO/TC154	EDI Sicherheit

Tabelle1: ISO-Kommittees, die sich mit Aspekten der Sicherheit beschäftigen.

ISO/IEC JTC1 zeichnet für OSI und die damit verbundenen Sicherheitskonzepte (IS 7498-2: OSI Sicherheitsarchitektur) verantwortlich. Darin werden die Sicherheitsdienste den Schichten des OSI7-Schichtenmodells zugeordnet. ISO/IEC JTC1 SC27 entwickelt Standards für Sicherheitsfunktionen und Evaluationskriterien. Insbesondere wird über die Anerkennung evaluierter Systeme nachgedacht. Bei all diesen Überlegungen werden jedoch die schon publizierten Kriterienkataloge von ITSEC und TCSEC berücksichtigt.

Darüber hinaus müssen die in den USA bei der NCSC in der Entwicklung befindlichen Evaluationskriterien als De-facto-Standard für sichere Systeme angesehen werden. Inzwischen wurde in den USA die Definition von Evaluationskriterien dem NIST, dem National Institute of Standards and Technology, übertragen. In einem ersten Entwurf wurden Erweiterungen der Klassen C2 und B1 des Orange Book vorgeschlagen und im Rahmen eines breit angelegten Reviewverfahrens angekündigt, an dem auch X/Open beteiligt werden soll.

SINIX und Standards für offene Systeme

Durch das Prinzip, Konformität zu anerkannten Standards zu bieten, sei es von Standardisierungsgremien, Konsortien oder anderen, wurde SINIX in Europa zum führenden offenen System im kommerziellen Bereich. SINIX bietet durch seine Standardkonformität größere Portabilität und Interoperabilität als andere Betriebssysteme. Wenn Anwendungssoftware durch die Konformität zu Standardschnittstellen portabel werden soll, ist SINIX die bevorzugte Referenzplattform. Wenn Anwendungssoftware aus Gründen der Portabilität nur Standardschnittstellen benutzt, kann sie problemlos auf SINIX-Systeme portiert werden. Wenn es ein System gibt, das erfolgreich in LANs und WANs über APIs und OSI, TCP/IP-, TRANSDATA, SNA, LAN-Manager, Novell- und X Protokolle kommunizieren und verteilte Verarbeitung einschließlich Transaktionsverarbeitung, unter Einbezug von PCs und Mainframe-Systemen, bieten kann, dann ist es SINIX.

Zu erwähnen ist an dieser Stelle auch, daß SINIX-Compiler den internationalen und europäischen Standards für Programmiersprachen entsprechen. Dies gilt ebenso für Kommunikationsstandards. SINIX ist wie die übrigen Systeme von Siemens Nixdorf der Masse in der Implementierung von Kommunikationsstandards, insbesondere der OSI-Standards, voraus. Konformität zu diesen Standards sowie Konnektivität und Interoperabilität werden von offiziell hierfür akkreditierten Testlabors geprüft und zertifiziert.

Siemens Nixdorf entwickelt seine Systeme standardkonform mit größter Beständigkeit und Sorgfalt. Standardkonformität bringt für Kunden und Softwarepartner einen berechenbaren Nutzen. Fehlende Standardkonformität, wie sie die Systeme einiger Anbieter aufweisen, kann durch keinerlei andere positive Eigenschaften ausgeglichen werden.

XPG3 PLUS: SINIX

Das Prüfzeichen XPG3 PLUS, das die XPG3-Konformität kennzeichnet, die über das minimale XPG3-Prüfzeichen hinausgeht, wurde nur von wenigen Produkten erzielt, allen voran von SINIX. Da die Bedingungen für das XPG3 -Prüfzeichen für jede Betriebssystemversion und jeden Prozessor, auf dem diese Betriebssystemversion ablauffähig ist, erfüllt werden müssen, hat SINIX das XPG3-PLUS-Prüfzeichen mehrmals erhalten. Die nachstehende Tabelle gibt eine Übersicht.

SINIX-Version	Prüfzeichen XPG3 PLUS seit
SINIX V5.22 auf MX300N/MX500	1. Juni 1990
SINIX-L V5.4 auf MX300 SINIX-D V5.41 auf WX200	3. Mai 1991
SINIX-ODT V1.5 auf WX200	4. Juli 1991
SINIX-M V5.40 auf MX500	6. Dezember 11991
SINIX-F V5.24 auf MX300N SINIX-H V5.24 auf MX 500N	11. Februar 1992
SINIX-N V5.41 auf RM400 SINIX-P V5.41 auf RM600	7. April 1992

Tabelle 2: SINIX hält die Hälfte aller Prüfzeichen XPG3 PLUS von X/Open.

Zur Zeit der Veröffentlichung des XPG4 im Oktober 1992 hatte Siemens Nixdorf bereits für zwölf Produkte das XPG4-Prüfzeichen beantragt. Inzwischen erhielt Siemens Nixdorf das erste XPG4-Base-Prüfzeichen für SINIX-L V5.41 auf MX300 und MX500 Rechnern. Dies war das dritte XPG4-Base-Prüfzeichen, das überhaupt von X/Open vergeben wurde. Danach folgten und folgen mit den Freigaben der relevanten SINIX-Produkte X/Open-Prüfzeichen für alle XPG4-Komponenten.

Siemens Nixdorf wird für alle XPG4-Komponenten und XPG4-Profile konforme SINIX-Produkte anbieten. Diese Strategie wird dadurch verdeutlicht, daß die Komponenten des XPG4 in das SINIX API aufgenommen werden. Auf diese Art werden sie auf allen strategischen SINIX-Linien garantiert. SINIX wird damit über das Profil-Branding des XPG4 hinausgehen.

XPG4-Profile haben gegenüber den XPG4-Komponenten eine zusätzliche, spezifische Konformitätsbedingung: „... ein einziges Quellprogramm ... sollte die Dienste der Portabilitäts-Schnittstelle aller Komponenten nutzen können." (... „a single source program ... shall be able to use the services provided by the Portability Interface of all those components.").

Der zusätzliche Vorteil eines XPG4-Profils liegt demnach in der Integration seiner Komponenten. Ein XPG4-Profil garantiert diese Integrationsbedingung jedoch nicht für Komponenten und Schnittstellen außerhalb des Profils. Diese Einschränkung entspringt einer Kompromißsituation bei X/Open. Vom Standpunkt des Anwenders aus ist sie untragbar. Es ist nicht vertretbar, daß bestimmte Schnittstellen von einem Programm benutzt werden können, die gleichzeitige Benutzung anderer Schnittstellen im gleichen System jedoch nicht sichergestellt wäre. Für SINIX gilt jedoch uneingeschränkt, daß alle Schnittstellenprodukte des SINIX API und alle SINIX-Produkte, für die die XPG4-Konformität realisiert ist, integriert sind.

Kein anderer Anbieter hat XPG4 und alle anderen Standards für offene Systeme so direkt und konsequent in seinen Produkten umgesetzt wie Siemens Nixdorf. Es wird insbesondere kein anderes Betriebssystem zu finden sein, das für all seine strategischen Versionen das X/Open-Warenzeichen für sämtliche XPG4-Komponenten und alle XPG4-Profile, und zwar mit vollständig integrierten Produkten, erhalten hat. SINIX ist damit die zukunftssicherste verfügbare Basis für Kundeninvestitionen in Anwendungssoftware, sowohl für Eigenentwicklungen als auch für Standard-Software.

Anhang B Abkürzungen

Die folgende Tabelle verzeichnet die wichtigsten der in diesem Buch verwendeten Abkürzungen in alphabethischer Reihenfolge sowie deren Auflösungen.

4GL	Fourth Generation Language
ABI	Application Binary Interface
ACL	Access Control Lists
ANSI	American National Standards Institute
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
BISON	Bull, ICL, Siemens, Olivetti und Nixdorf
BMS	Broadcast Message Server
BSD	Berkeley Software Distribution
CAD	Computer Aided Design
CAE	Common Applications Environment
CASE	Computer-Aided Softwre Engineering
CBT	Computer Based Training
CCITT	Consulting Committee on International Telegraphy and Telephony
CDS	Cell Directory Service

CISC	Complex Instruction Set Computer
CLI	Command Line Interface
CMIS	Common Management Information Service
CMPI	Common Management Information Protocol
CORBA	Common Object Request Broker Architecture
COSE	Common Open Software Environment
CPI-C	Common Programming Interface for Communication
DARPA	Defense Advanced Research Projects Agency
DCE	Distributed Computing Environment
DDI/DKI	Device Driver Interface, Driver-Kernel Interface
DFS	Distributed File System
DIA	Display Industry Association
DIN	Deutsche Industrienorm
DNS	Domain Name Service
DO	Data Object
DRDA	Distributed Relational Database Architecture
DSA	Direcory System Agents
DTS	Distributed Time Service
DUA	Directory User Agents
ECMA	European Computer Manufacturer´s Association
ELF	ist ein Objektformat in UNIX/SINIX
ERMS	Enterprise Repository Management System
EWOS	European Workshop for Open Systems

FACE	Framed Access Command Environment
FML	Form and Menu Language
FMLI	Form and Menu Language Interpreter
FTP	File Transfer Protocol
GDS	Global Directory Service
GUI	Graphical User Interface
IANA	Internet Assigned Numbers Authority
IDL	Interface Definition Language
IEC	International Electrotechnical Commitee
IEEE	Institute for Electrical and Electronics Engineers
INMS	Interface Network Management Service
ISO	International Standards Organization
JTC1	Joint Technical Committee
KDCS	Kompatible Datenkommunikations-Schnittstelle
LAN	Local Area Network
MAN	Metropolitan Area Networks
MHB	Siemens Nixdorf Methodenhandbuch
MIB	Management Information Base
MIMD	multiple instruction multiple data
MS	Managing System
MS-DOS	Microsoft Disk Operating System
NEA	Network Environment Architecture
NCSC	National Computer Security Center

NFS	Network File System
NIST	National Institute for Standards and Technology
NLS	Native Language System
NMF	Network Management Forum
NMS	Network Management Services
NSA	National Security Agency
OA&M	Operation, Administration and Maintenance
OED	Object Editor
OLTP	On Line Transaction Processing
OMG	Object Management Group
ORB	Object Request Broker
OSCE	Open System CASE Environment
OSF	Open Software Foundation
OSI	Open Systems Interconnection
PIN	Personal Identification Number
RCS	Revision Control System
RFT	Request for Technology
RISC	Reduced Instruction Set Computer
RPC	Remote Procedure Calls
SCCS	Source Code Control System
SCSI	Small Computer System Interface
SD	SINIX/Windows Development
SMI	Structure of Management Information

SIMD	single instruction multiple data
SNA	System Network Architecture
SPOOL	Simultaneous Peripheral Operations On Line
SQL	Structured Query Language
SUE	SINIX/Windows User Interface
SVR4	UNIX System V Release 4
TAC	Transaktionscode
TCP/IP	Transmission Communication Protocol/Internet Protocol
TCSEC	Trusted Computer System Evaluation Criteria
TRANSVIEW-DAME	TRANSVIEW Dynamic Administration Management Environment
TRANSVIEW-DSM	TRANSVIEW Distributed Systems Management
TRANSVIEW-SNMP	TRANSVIEW Simple Mail Transfer Protocol
UI	UNIX International
UIL	User Interface Language
UDP	User Datagram Protocol
ULS	Universal Language System
USL	UNIX System Laboratories
VFS	Virtual File System
VxFS	VERITAS FILE System
WAN	Wide Area Networks
WYSYWYG	What you see is what you get
XPG	X/Open Portability Guide

Anhang C Literatur

[1] Booch, Grady,
Object-Oriented Design,
Benjamin/Cummings, Menlo Park, Califonia, 1990.

[2] Ellis, Margaret and Stroustrup, Bjarne
The Annotated C++ Reference Manual,
Addison-Weseley, Reading, Massachusetts, 1990.

[3] Hughes, U.S. Department of Defense,
Composite Software Error History.

[4] Kernighan, Brian. W., and Ritchie, Dennis. M.,
The C Programming Language,
Prentice Hall, Englewood Cliffs, N. J., 1978.

dt: Programmieren in C,
Carl Hanser, München, 1983.

[5] Rose, Marshall T. ,
The Open Book,
Prentice Hall, Englewood Cliffs, N.J. 1990.

[6] Slonim, J., Schonbach, A., Bauer, M, MacRae, L., and Thomas, K.,
Building An Open System,
Van Nostrand Reinhold Company, New York, 1987.

Index

Ziel dieses Buches ist es, einen umfassenden Überblick über SINIX zu geben und zu zeigen, warum gerade SINIX und seine Komponenten in besonderem Maß für den kommerziellen Einsatz geeignet sind. Den Hintergrund bildet die breite Palette der SINIX-Produkte: das Betriebssystem SINIX selbst, Werkzeuge und Arbeitsumgebungen, aber auch Anwendungsprogramme, die von SINIX unterstützt werden.

Inhaltsübersicht: SINIX ist UNIX, ein offenes System, ideal für den kommerziellen Einsatz - Das Betriebssystem SINIX - Kommunikation und Netzwerke - Benutzerumgebung - Entwicklungsumgebung - Systemverwaltung und Netzmanagement - Verteilte Verarbeitung - SINIX-Anwendungen im kommerziellen Großeinsatz - Verfügbarkeit, Sicherheit und Qualität in SINIX - Zukünftige Entwicklungen



Siemens Nixdorf
Informationssysteme AG
Bestell-Nr./Order No.
U21497-J-Z145-1
2890 AG 7930.5(4800)



ISBN 3-446-17647-0